**UNIVERSIDADE PAULISTA**

**MARCOS PAULO FRANCISCO VAZ F09BEG0**

**MATHEUS DOS SANTOS SILVA N590785**

**MATHEUS OLIVEIRA DE MORAES N477GA2**

**SAMUEL ARAUJO DE SOUZA F30AJG4**

**OPERAÇÕES DE CERCO E ACOMPANHAMENTO DA POLÍCIA MILITAR:**

Desenvolvimento do escopo de um sistema gráfico de tempo real

**SÃO PAULO**

**2023**

**MARCOS PAULO FRANCISCO VAZ F09BEG0**

**MATHEUS DOS SANTOS SILVA N590785**

**MATHEUS OLIVEIRA DE MORAES N477GA2**

**SAMUEL ARAUJO DE SOUZA F30AJG4**

**OPERAÇÕES DE CERCO E ACOMPANHAMENTO DA POLÍCIA MILITAR:**

Desenvolvimento do escopo de um sistema gráfico de tempo real

Atividade prática supervisionada do 7º semestre do curso de Ciência da Computação apresentado à Universidade Paulista.

Professor: Paulo Freitas

**SÃO PAULO**

**2023**

**LISTA DE FIGURAS**

[Figura 1 - Método em cascata usando o diagrama de Gantt 8](#_Toc135948077)

[Figura 2 - Relações entre expectativa e percepção na gestão de escopo 13](#_Toc135948078)

[Figura 3 - Howard Stark, no filme Iron Man (2018), aparecendo em gravação para seu filho citando uma tecnologia inviável de ser construída em seu tempo devido a inviabilidade técnica 22](#_Toc135948079)

[Figura 4 – Reator *arc:* um dispositivo inviável de ser construído na época do Howard Stark, alimentado por uma substância chamada paládio, que é capaz de gerar uma poderosa fonte de energia limpa e sustentar o traje do Homem de Ferro 22](#_Toc135948080)

[Figura 5 - Ilustração de entrevista com cliente para elicitação de requisitos 28](#_Toc135948081)

[Figura 6 - Ilustração de uma reunião de brainstorming 30](#_Toc135948082)

[Figura 7 - Sessão de brainwriting conduzida por Murilo Gun, fundador da *Keep Learning School* 33](#_Toc135948083)

[Figura 8 - Exemplo de wireframes 35](#_Toc135948084)

[Figura 9 - Figma como ferramenta para desenvolver protótipos de alto nível 37](#_Toc135948085)

[Figura 10 - Adobe XD como ferramenta para desenvolver protótipos de alto nível 38](#_Toc135948086)

[Figura 11 - Diagrama de casos de uso para um elevador 47](#_Toc135948087)

[Figura 12 - Diagrama de classe de uma unidade de manutenção de elevador 49](#_Toc135948088)

[Figura 13 - Tipos de relações entre diagramas de classe e suas representações gráficas 50](#_Toc135948089)

[Figura 14 - Diagrama de classes para um sistema de clínica dentaria 51](#_Toc135948090)

[Figura 15 - Diagrama de sequência para um sistema bancário 54](#_Toc135948091)

[Figura 16 - Linhas de vida no diagrama de sequência 55](#_Toc135948092)

[Figura 17 - Atores no diagrama de sequência 55](#_Toc135948093)

[Figura 18 - Barras de ativação no diagrama de sequência 56](#_Toc135948094)

[Figura 19 - Seta para representação de mensagem síncrona no diagrama de sequência 56](#_Toc135948095)

[Figura 20 - Seta para representação de mensagem assíncrona no diagrama de sequência 57](#_Toc135948096)

[Figura 21 - Seta para representação de uma mensagem de retorno no diagrama de sequência 57](#_Toc135948097)

[Figura 22 - Seta indicando a criação de um objeto participante no diagrama de sequência 58](#_Toc135948098)

[Figura 23 - Um símbolo "X" indicando o fim da participação de um objeto no diagrama de sequência 59](#_Toc135948099)

[Figura 24 - Uma seta que aponta para o próprio objeto no diagrama de sequência 60](#_Toc135948100)

[Figura 25 - Atividades e setas no diagrama de atividades 61](#_Toc135948101)

[Figura 26 - Símbolo de decisão que ramifica o fluxo de atividades no diagrama de atividades 62](#_Toc135948102)

[Figura 27 - Fork e join representando o paralelismo e concorrência no diagrama de atividades 63](#_Toc135948103)

[Figura 28 - Diagrama de estados para um sistema de criação de viagens 65](#_Toc135948104)

[Figura 29 - Caso de uso para cadastros de equipes de policiais e operadores do COPOM 75](#_Toc135948105)

[Figura 30 - Casos de uso para o sistema Comunicador Inter-agentes 75](#_Toc135948106)

[Figura 31 - Diagrama de classe do Comunicador Inter-agente 75](#_Toc135948107)

[Figura 32 - Rascunho da interface de cadastro de policiais e operadores do COPOM 79](#_Toc135948108)

[Figura 33 – Rascunho da interface de login desktop e mobile 79](#_Toc135948109)

[Figura 34 - Telas de patrulhamento, início de acompanhamento e solicitação de apoio 80](#_Toc135948110)

**SUMÁRIO**

[1 OBJETIVO 7](#_Toc135948019)

[2 INTRODUÇÃO 8](#_Toc135948020)

[3 FUNDAMENTAÇÃO TEÓRICA 11](#_Toc135948021)

[3.1 Escopo de um projeto 11](#_Toc135948022)

[3.2 Engenharia 15](#_Toc135948023)

[3.2.1 Um breve histórico da engenharia 16](#_Toc135948024)

[3.2.2 Definições formais de engenharia 17](#_Toc135948025)

[3.3 Engenharia de requisitos e requisitos de software 18](#_Toc135948026)

[3.3.1 Estudo de viabilidade 20](#_Toc135948027)

[3.3.2 Elicitação de requisitos 23](#_Toc135948028)

[3.3.2.1 Questionários 26](#_Toc135948029)

[3.3.2.2 Entrevistas 28](#_Toc135948030)

[3.3.2.3 Dinâmica de grupo 30](#_Toc135948031)

[3.3.2.4 Prototipação 34](#_Toc135948032)

[3.3.2.5 Etnografia 38](#_Toc135948033)

[3.3.3 Análise de requisitos 39](#_Toc135948034)

[3.3.4 Validação 41](#_Toc135948035)

[3.4 Modelagem gráfica 44](#_Toc135948036)

[3.4.1 Caso de uso 46](#_Toc135948037)

[3.4.2 Diagrama de classe 48](#_Toc135948038)

[3.4.3 Diagrama de sequência 53](#_Toc135948039)

[3.4.4 Diagramas de atividade 61](#_Toc135948040)

[3.4.5 Diagramas de estado 64](#_Toc135948041)

[4 DOCUMENTO DE REQUISITO: COMUNICADOR INTER-AGENTES 66](#_Toc135948042)

[4.1 Ficha técnica 66](#_Toc135948043)

[4.1.1 Equipe responsável pela elaboração 66](#_Toc135948044)

[4.1.2 Público-alvo 66](#_Toc135948045)

[4.2 Introdução 66](#_Toc135948046)

[4.2.1 Visão geral deste documento 67](#_Toc135948047)

[4.2.2 Convenções, termos e abreviações 67](#_Toc135948048)

[4.2.2.1 Identificação dos requisitos 67](#_Toc135948049)

[4.2.2.2 Prioridade do requisito 68](#_Toc135948050)

[4.3 Descrição geral do sistema 68](#_Toc135948051)

[4.3.1 Abrangência e sistemas relacionados 69](#_Toc135948052)

[4.3.2 Descrição dos usuários 70](#_Toc135948053)

[4.4 Requisitos funcionais (casos de uso) 71](#_Toc135948054)

[4.4.1 [RF001] Cadastrar equipe de campo 71](#_Toc135948055)

[4.4.2 [RF002] Cadastrar operador da COPOM 72](#_Toc135948056)

[4.4.3 [RF003] Iniciar operações 72](#_Toc135948057)

[4.4.4 [RF004] Iniciar patrulhamento 73](#_Toc135948058)

[4.4.5 [RF005] Iniciar acompanhamento 73](#_Toc135948059)

[4.4.6 [RF006] Solicitar apoio 74](#_Toc135948060)

[4.4.7 [RF007] Acompanhar livestream 74](#_Toc135948061)

[4.5 Requisitos não funcionais 76](#_Toc135948062)

[4.5.1 [NF001] Interface de patrulhamento minimalista 76](#_Toc135948063)

[4.5.2 [NF002] Botão para acionar “iniciar acompanhamento” 76](#_Toc135948064)

[4.5.3 [NF003] Usar voz para acionar “iniciar acompanhamento” 77](#_Toc135948065)

[4.5.4 [NF005] Utilização do 5G para comunicação em tempo real 77](#_Toc135948066)

[4.5.5 [NF006] Manter a sessão em regiões sem cobertura de sinal 77](#_Toc135948067)

[4.5.6 [NF007] Segurança e prevenção de interceptação de mensagens 77](#_Toc135948068)

[4.5.7 [NF008] Quantidade de policiais no apoio 78](#_Toc135948069)

[4.5.8 [NF009] Baixo tempo de resposta 78](#_Toc135948070)

[4.6 Descrição da interface com o usuário 79](#_Toc135948071)

[4.6.1 [IN001] Cadastro de equipes e operadores do COPOM 79](#_Toc135948072)

[4.6.2 [IN002] Login mobile e desktop 79](#_Toc135948073)

[4.6.3 [IN003] Inicio de patrulhamento e solicitação de apoio 80](#_Toc135948074)

[5 CONCLUSÃO 82](#_Toc135948075)

[REFERÊNCIAS 83](#_Toc135948076)

# OBJETIVO

O objetivo deste trabalho é realizar uma pesquisa exploratória sobre os processos de Engenharia de Requisitos, com ênfase na elicitação e validação, visando fornecer uma compreensão aprofundada dessas etapas essenciais no desenvolvimento de software.

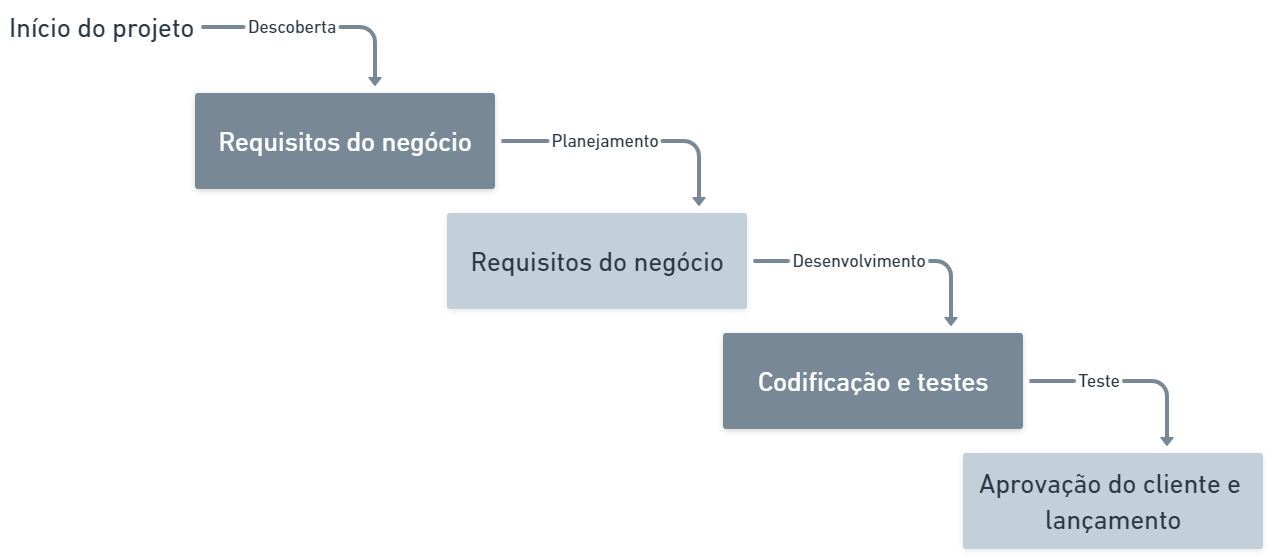
Além disso, como objetivo técnico, busca-se elaborar um documento de requisitos detalhado para um projeto de software real – Comunicador Inter-agentes –, aplicando as melhores práticas e técnicas pesquisadas.

# INTRODUÇÃO

Havia uma época em que o desenvolvimento de software seguia um modelo rígido, conhecido como modelo cascata (figura 1). Nesse modelo, desenhavam-se fluxos que mostrava cada uma das fases do projeto como uma cascata. Esses fluxos se chamam “diagramas de Gantt”, em homenagem a Henry Gantt, que os desenvolveu.

Os clientes se envolviam no início do projeto e definiam o escopo de uma só vez, descrevendo em detalhes todos os requisitos e funcionalidades que desejavam ver no produto.

Figura 1 - Método em cascata usando o diagrama de Gantt



Fonte: Sutherland, 2014

Empolgados com a oportunidade de terem suas necessidades atendidas, os clientes compartilhavam suas visões e expectativas com a equipe de desenvolvimento. Os desenvolvedores, por sua vez, trabalhavam incansavelmente durante meses para construir o software de acordo com as especificações fornecidas. Muitos problemas de estimativa (prazo e dinheiro) aconteciam. Mas, sempre existiam equipes que conseguiam concluir o projeto.

No entanto, ironicamente, quando o projeto finalmente era entregue aos clientes, muitas vezes havia uma sensação de descontentamento. O software não correspondia às suas expectativas e necessidades, mesmo que os desenvolvedores tivessem seguido todas as especificações do escopo original.

Por que isso aconteceu? Um planejamento não tinha sido realizado? É muito comum uma frase no meio do desenvolvimento de software “o cliente não sabe o que quer”. Alguns ainda vão um pouco na contramão dessa frase e diz “o cliente sabe muito bem o que ele quer, só não sabe dizer direito” (CARDOSO, 2020).

Durante o desenvolvimento, os clientes percebiam que suas necessidades haviam mudado ao longo do tempo ou que não haviam expressado completamente suas expectativas desde o início. No entanto, como o planejamento já havia sido feito, não há espaço para um *feedback* e mudança.

Tradicionalmente, a gerência quer duas coisas em qualquer projeto: controle e previsibilidade. Meses de esforço para o planejamento de todos os detalhes, para que não haja nenhum erro, para que o orçamento não estoure, e para que tudo seja entregue no prazo.

Em vez disso, esse tipo de planejamento gera frustação porque as pessoas não conseguem o que realmente queriam. Os projetos atrasam e estouram o orçamento e, em muitos casos, acabam não dando certo. Isso é verdadeiro principalmente nos casos que envolvem equipes criativas trabalhando em algo novo. Na maioria das vezes, a gerência não toma conhecimento do caminho em direção ao fracasso, até que milhões de dólares e milhares de horas tenham sido investidos em vão (SUTHERLAND, 2014).

Essa abordagem inflexível de definir o escopo uma única vez não permitia ajustes durante o processo de desenvolvimento, tornando difícil responder às mudanças de requisitos e às novas perspectivas dos clientes. O resultado era um produto que não atendia às suas necessidades e não fornecia o valor esperado.

Nesse contexto, esta pesquisa visa mostrar que a coleta e especificação de requisitos é muito importante, visto que esses são muito importantes para o negócio. Também, que devido a natureza dos requisitos de um projeto de software, é necessário validar constantemente se eles continuam alinhados com os objetivos uma vez estabelecidos

Na seção III, será abordada a fundamentação teórica relacionada ao escopo de um projeto de software. Serão discutidos os elementos essenciais para a definição do escopo, incluindo a identificação e o estabelecimento de limites claros, a definição dos objetivos e a delimitação das funcionalidades a serem desenvolvidas. Além disso, serão explorados os métodos e técnicas utilizados para a elicitação de requisitos, bem como a importância da comunicação eficaz entre os stakeholders do projeto.

Na seção IV, será apresentada uma documentação de requisitos para um projeto de software real. Os conceitos estabelecidos na sessão anterior serão colocados em prática.

# FUNDAMENTAÇÃO TEÓRICA

## Escopo de um projeto

Segundo o PMBoK (apud FREITAS), gestão do escopo do projeto são “os processos necessários para assegurar que o projeto inclui todo o trabalho necessário, e apenas o necessário, para terminar o projeto com sucesso”.

O *Chaos Report* identifica a gestão de escopo como um dos principais fatores críticos de sucesso em projetos. O Chaos Report é um estudo realizado pelo *Standish Group*,que analisa a taxa de sucesso de projetos de software e identifica os principais desafios e fatores que influenciam o seu resultado.

De acordo com o relatório, a gestão de escopo adequada é essencial para o sucesso de um projeto. Quando o escopo não é adequadamente definido, controlado e gerenciado, as chances de o projeto enfrentar problemas aumentam consideravelmente. Esses problemas podem incluir atrasos, orçamentos estourados, entregas insatisfatórias e até mesmo o fracasso completo do projeto. A seguir, a tabela 1 mostra o sucesso e falhas em projetos de 2011 a 2015:

Tabela 1 - A resolução tradicional de todos os projetos de software do ano de 2011 a 2015 no banco de dados CHAOS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 2011 | 2012 | 2013 | 2014 | 2015 |
| Sucesso | 39% | 37% | 41% | 36% | 36% |
| Challenged | 39% | 46% | 40% | 47% | 45% |
| Falhou | 22% | 17% | 19% | 17% | 19% |

Fonte: Standish Group, 2015[[1]](#footnote-1)

Uma gestão de escopo eficaz envolve a definição clara dos objetivos, resultados esperados e limites do projeto, bem como a comunicação e o alinhamento com as partes interessadas. Isso permite que a equipe do projeto tenha uma visão clara do que precisa ser entregue, evitando mudanças e expansões desnecessárias que possam comprometer o sucesso do projeto.

Ao definir um escopo realista e alcançável, é possível estabelecer expectativas claras para todos os envolvidos, alinhar recursos e esforços adequadamente e mitigar riscos. Além disso, uma gestão de escopo eficiente inclui o monitoramento contínuo do escopo ao longo do ciclo de vida do projeto, garantindo que as mudanças sejam avaliadas, aprovadas e gerenciadas de maneira adequada.

Dessa forma, a gestão de escopo desempenha um papel fundamental na garantia do sucesso de um projeto, minimizando os riscos associados a mudanças e desvios não controlados. É por esse motivo que o Chaos Report e outros estudos destacam a importância desse aspecto na gestão de projetos de software.

Na gestão de escopo, é fundamental gerenciar as expectativas das partes interessadas envolvidas no projeto. Gerenciar expectativas significa garantir que todas as partes tenham uma compreensão clara e realista do que será entregue pelo projeto, dos prazos envolvidos, dos recursos disponíveis e das restrições associadas.

Gerenciar expectativas é uma habilidade valiosa em todos os aspectos da vida, não apenas na gestão de projetos. Em 2023, no *YouTube* brasileiro, aconteceu o “episódio Raluca”, uma briga na internet onde uma das partes, Raluca, prometeu criar um conteúdo que “destruiria” outra a parte. Foi alimentada a expectativa do público de forma que, quando lançado o conteúdo, não correspondeu ao que se esperava. Houve uma decepção generalizada.

Na gestão de expectativas, é importante entender dois conceitos relacionados: percepção e expectativa.

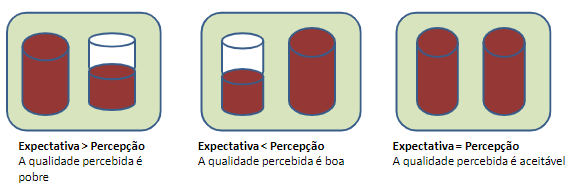
A percepção refere-se à maneira como uma pessoa interpreta e compreende uma situação, evento ou experiência. A percepção é subjetiva e pode variar de pessoa para pessoa, pois é influenciada por fatores como experiências passadas, valores, crenças e emoções. A percepção molda a forma como uma pessoa vê o mundo ao seu redor e influencia suas expectativas em relação a eventos futuros.

A expectativa é uma crença ou suposição sobre o que é provável que aconteça no futuro, com base na percepção atual e em experiências anteriores. As expectativas podem ser conscientes ou inconscientes e podem ser influenciadas por uma variedade de fatores, como informações disponíveis, comunicação, crenças pessoais e expectativas sociais. As expectativas podem afetar a forma como as pessoas se comportam, tomam decisões e reagem a eventos.

Na gestão de expectativas, é importante considerar tanto a percepção quanto a expectativa das partes envolvidas. Isso envolve entender como cada pessoa percebe a situação e quais são as suas expectativas em relação a um determinado resultado. Compreender e abordar esses aspectos pode ajudar a alinhar as expectativas com a realidade, evitar mal-entendidos e facilitar a comunicação efetiva.

A figura 2 a seguir mostra o balanço entre esses dois conceitos:

Figura 2 - Relações entre expectativa e percepção na gestão de escopo



Fonte: Freitas, 2022

Gerenciar percepções e expectativas requer uma abordagem cuidadosa e empática. Envolve ouvir ativamente as preocupações e expectativas das pessoas envolvidas, fornecer informações claras e realistas, comunicar-se de maneira transparente e honesta, e adaptar-se quando necessário para atender às expectativas razoáveis.

Ao abordar percepções e expectativas na gestão de projetos, relacionamentos pessoais ou outras áreas da vida, é importante lembrar que as percepções e expectativas podem mudar ao longo do tempo. Portanto, é fundamental manter uma comunicação aberta, estar disposto a revisar e reavaliar as expectativas conforme necessário, e trabalhar em colaboração para alcançar resultados que atendam às necessidades e expectativas de todas as partes envolvidas.

Mesclado a expectativa e satisfação, existe um terceiro conceito que é resultado desses dois: a satisfação, que é o sentimento de contentamento ou realização que uma pessoa experimenta em relação a uma situação ou evento. Ela surge quando a percepção do resultado alcançado está alinhada ou supera as expectativas iniciais.

A relação entre esses conceitos pode ser descrita da seguinte maneira:

* Expectativa superada: Quando o resultado de uma situação ou evento excede a expectativa de uma pessoa, a probabilidade de sentir satisfação é alta. Nesse caso, a percepção do resultado é positiva e alinhada com a expectativa inicial ou até mesmo superior a ela, resultando em uma sensação de satisfação.
* Expectativa atendida: Quando o resultado de uma situação ou evento está em linha com as expectativas de uma pessoa, isso geralmente leva a uma sensação de satisfação. A percepção do resultado é consistente com a expectativa, e a pessoa sente que suas necessidades e desejos foram cumpridos.
* Expectativa não atendida: Se o resultado de uma situação ou evento não atende às expectativas de uma pessoa, isso pode levar à insatisfação. A percepção do resultado pode ser negativa, pois não corresponde ao que a pessoa esperava ou desejava.

É importante destacar que a satisfação não depende apenas de atender ou superar as expectativas, mas também da percepção individual do resultado. Duas pessoas podem ter expectativas diferentes para a mesma situação e, consequentemente, experimentar diferentes níveis de satisfação com o mesmo resultado.

Na gestão de projetos, relacionamentos ou qualquer outra área da vida, entender as expectativas e percepções das pessoas envolvidas é fundamental para alcançar a satisfação. Comunicar de forma clara, estabelecer expectativas realistas e trabalhar em direção a resultados que atendam às percepções e expectativas das partes interessadas pode contribuir para a satisfação geral e o sucesso das relações e empreendimento.

Os processos de planejamento da gestão de escopo fornecem as diretrizes e estratégias para definir, documentar e gerenciar o escopo de um projeto. Eles envolvem a identificação das atividades, produtos, serviços e resultados que devem ser entregues para satisfazer os objetivos do projeto.

A seguir, estão os principais processos de planejamento da gestão de escopo:

1. Definir o escopo: Nesse processo, o objetivo é estabelecer uma declaração clara e concisa do escopo do projeto, descrevendo os limites, as entregas, as restrições e as premissas. Isso inclui identificar os objetivos, os requisitos e as expectativas das partes interessadas, além de estabelecer critérios de aceitação;
2. Desenvolver a declaração do escopo: Aqui, é criada uma declaração detalhada do escopo do projeto, documentando de forma completa e precisa os resultados, os produtos e os serviços a serem entregues. Essa declaração inclui também exclusões e limitações, estabelecendo claramente o que está dentro e o que está fora do escopo do projeto;
3. Criar a EAP (Estrutura Analítica do Projeto): A EAP é uma representação hierárquica de todas as entregas e atividades do projeto. Nesse processo, as principais entregas são decompostas em pacotes de trabalho menores e mais gerenciáveis. A EAP ajuda a organizar e visualizar o escopo do projeto, facilitando o planejamento, a estimativa de recursos e o controle.

Aqui, processos relacionados ao monitoramento e controle

1. Verificar o escopo: Esse processo envolve revisar formalmente o trabalho concluído para garantir que ele esteja em conformidade com os requisitos e expectativas definidos. A verificação do escopo é realizada em conjunto com as partes interessadas relevantes, para confirmar se as entregas estão completas, corretas e aprovadas;
2. Controlar o escopo: Esse processo se concentra em monitorar e controlar as mudanças no escopo ao longo do ciclo de vida do projeto. Ele envolve avaliar as solicitações de mudança, analisar os impactos no escopo, nas entregas e no cronograma, e tomar decisões sobre a aprovação ou rejeição das mudanças. O controle do escopo garante que o projeto permaneça dentro dos limites definidos e ajuda a evitar a expansão descontrolada do escopo.

Esses processos de planejamento da gestão de escopo são interligados e geralmente ocorrem de forma iterativa, à medida que mais informações são obtidas e a compreensão do escopo se aprofunda ao longo do projeto. Eles fornecem a estrutura necessária para definir, documentar e controlar efetivamente o escopo do projeto, permitindo uma gestão adequada do trabalho necessário para o sucesso do projeto.

## Engenharia

Engenharia é uma disciplina que abrange várias áreas e disciplinas do conhecimento, como química, computação, eletricidade, software, meio ambiente e muitas outras.

Historicamente, a engenharia tem sido associada a grandes figuras da história da humanidade, como Leonardo da Vinci, conhecido por suas contribuições na engenharia mecânica e civil, e Nikola Tesla, famoso por suas inovações na área da eletricidade e do eletromagnetismo. Esses visionários e muitos outros engenheiros notáveis deixaram um legado significativo por meio de suas contribuições para o desenvolvimento de tecnologias revolucionárias e avanços científicos.

Na cultura popular, a figura do engenheiro frequentemente é retratada como um herói mecânico que constrói suas armas, inspirando pessoas a buscar sua vocação e explorar o campo da engenharia. Esses heróis engenheiros, tanto fictícios quanto reais, ajudam a transmitir a importância da engenharia na resolução de problemas complexos e na construção de um futuro melhor.

Engenharia de requisitos é mais uma palavra que entra para esse conjunto. Contudo, o que é engenharia de fato?

### Um breve histórico da engenharia

De fato, a origem etimológica da palavra "engenharia" remonta ao latim "*ingenium*", que significa "aquele que inventa" ou "mente criativa". Essa raiz etimológica reflete a essência da engenharia como uma disciplina que envolve o processo de criar, projetar e desenvolver soluções inovadoras para desafios técnicos e problemas complexos.

A engenharia está intrinsecamente ligada a características como criação, projeto, design, processos, artefatos e ferramentas.

A raiz etimologia da palavra vem do latim *ingenium*, e significa algo como “aquele que inventa”. Algumas palavras associadas: *ingeniósus* (engenhosidade) e *ingenii* (criatividade). Algumas características, normalmente relacionadas à engenharia, podem melhorar nosso entendimento: criação, projeto, design, processos, artefatos, ferramentas (L. ANDERSON, 2019).

O artigo intitulado "Programas de Tecnologia Industrial: Relação com Engenharia e Gestão" publicado em 1990 por W. Sinn et al. (p. 5) apresenta uma narrativa episódica sobre a história da engenharia. Embora a engenharia possua uma longa história que remonta às primeiras civilizações, onde foram desenvolvidos canais de irrigação, estruturas de saneamento e outras realizações notáveis, é no século 19, com a Revolução Industrial, que a interpretação moderna da engenharia começa a emergir.

Até então, a engenharia era comumente definida como "a arte de manipular os recursos da natureza em benefício da humanidade". Nessa época, os responsáveis por promover o progresso tecnológico eram os artesãos e engenheiros. No entanto, a engenharia era predominantemente vista como uma arte, em vez de um resultado de processos analíticos e científicos. Uma citação notável de Theodore von Kármán, um engenheiro americano do século 19, exemplifica essa perspectiva artística da engenharia: "um cientista estuda o que é, enquanto os engenheiros criam o que nunca foi" (Anderson, 2019).

À medida que o século 20 avançava, a ciência começou a ser cada vez mais incorporada às definições de engenharia. Nesse ponto, o espectro de profissionais envolvidos nos avanços tecnológicos expandiu-se para incluir cientistas, juntamente com os engenheiros. Essa integração entre ciência e engenharia trouxe uma nova dimensão para o campo, permitindo abordagens mais analíticas e fundamentadas em evidências para resolver desafios complexos.

Com o tempo, a engenharia evoluiu significativamente, abrangendo diversas disciplinas, como engenharia mecânica, civil, elétrica, de computação e muitas outras. A introdução de metodologias sistemáticas, ferramentas avançadas e princípios científicos levou ao desenvolvimento de novos produtos, tecnologias e infraestruturas que moldaram a sociedade moderna.

É importante reconhecer que a história da engenharia é contínua e em constante evolução. À medida que novas descobertas científicas são feitas e os desafios do mundo contemporâneo são enfrentados, a engenharia continua a se adaptar e aprimorar suas práticas para atender às necessidades da sociedade em constante mudança.

### Definições formais de engenharia

A *Accreditation Board for Engineering and Technology* (ABET) define a engenharia como "a profissão na qual o conhecimento da matemática e das ciências naturais é aplicado para criar formas economicamente viáveis de utilizar os materiais e forças da natureza em benefício da raça humana" (SINN et al., 1990, p. 5, tradução nossa). Nessa definição, é importante destacar a palavra "economicamente". Isso significa que os engenheiros devem considerar a eficiência e a otimização dos recursos ao projetar soluções, buscando maneiras de utilizar os materiais e as forças da natureza de forma econômica e sustentável.

Como já foi visto, o guia *Project Management Body of Knowledge* (PMBoK) do *Project Management Institute* (PMI, 2013) define o gerenciamento de escopo como um conjunto de processos que visam garantir que o projeto inclua todo o trabalho necessário, e apenas o necessário, para sua conclusão bem-sucedida. Nessa definição, destaca-se a importância de determinar de forma precisa e abrangente o escopo do projeto, evitando a inclusão de elementos desnecessários que possam comprometer a eficiência e o sucesso do projeto. A abordagem de gerenciamento de escopo busca garantir que as atividades, entregas e requisitos estejam claramente definidos, alinhados com os objetivos do projeto e controlados ao longo do ciclo de vida do projeto.

Além disso, o *National Assessment Governing Board* (NAGB) define a engenharia como "uma abordagem sistemática e, frequentemente, iterativa para desenvolver objetos, processos e sistemas que atendam às necessidades e desejos humanos". Nessa definição, destaca-se a palavra "iterativa", que se refere ao fato de que o processo de engenharia muitas vezes envolve um ciclo de desenvolvimento contínuo, no qual as soluções são aprimoradas e refinadas ao longo do tempo com base no feedback e nas necessidades em constante evolução.

Essas duas definições destacam aspectos importantes da engenharia. A abordagem econômica enfatiza a necessidade de considerar a eficiência e a sustentabilidade no uso dos recursos naturais, buscando soluções viáveis e economicamente conscientes. Já a abordagem iterativa ressalta a natureza evolutiva do trabalho dos engenheiros, que estão constantemente procurando melhorar e aprimorar as soluções existentes para atender às necessidades e desejos humanos em constante mudança.

Essas palavras destacadas, "economicamente" e "iterativa", serão exploradas em detalhes posteriormente a fim de compreender melhor seu significado e importância no contexto da engenharia e do gerenciamento de escopo de projetos.

## Engenharia de requisitos e requisitos de software

A engenharia está intimamente relacionada aos requisitos em um projeto. Antes de explorarmos essa relação, é importante compreender o que são requisitos. O termo "requisito" tem sua origem no latim "*requīsitus*", que significa algo iniludível ou imprescindível para o desenvolvimento de algo (CONCEITO.DE, 2023).

Dessa forma, a engenharia de requisitos envolve os processos sistemáticos necessários para estabelecer os requisitos de um determinado projeto (SUTCLIFFE, 2023). Existem várias definições de engenharia de requisitos na academia, mas todas compartilham a ideia fundamental de que requisitos envolvem descobrir o que as pessoas desejam ou necessitam em relação a um projeto específico, ou seja, qual é o resultado esperado, o "*output*".

Os requisitos em um projeto podem ser entendidos como as funcionalidades, características, restrições e objetivos que o produto ou sistema final deve atender para atender às necessidades e expectativas dos stakeholders. Os requisitos podem ser divididos em requisitos funcionais, que descrevem as funcionalidades e comportamentos esperados do produto, e requisitos não funcionais, que abrangem aspectos como desempenho, usabilidade, segurança e outros critérios de qualidade.

A engenharia de requisitos desempenha um papel crucial na definição e gestão dos requisitos ao longo do ciclo de vida de um projeto. Ela envolve técnicas e atividades para identificar, analisar, documentar, validar e verificar os requisitos do projeto. Isso inclui a interação com os stakeholders, a compreensão de suas necessidades e expectativas, a tradução dessas informações em requisitos claros e a garantia de que os requisitos estão alinhados com as metas e objetivos do projeto.

A relação entre a engenharia e os requisitos está no fato de que a engenharia é responsável por traduzir as necessidades e expectativas dos stakeholders em soluções técnicas concretas. Os engenheiros devem entender completamente os requisitos para projetar e desenvolver produtos e sistemas que atendam às especificações e satisfaçam as necessidades dos usuários finais.

Portanto, a engenharia e os requisitos são interdependentes. A engenharia utiliza os requisitos como base para o desenvolvimento e a tomada de decisões técnicas, enquanto a engenharia de requisitos utiliza os princípios e técnicas da engenharia para identificar, analisar e gerenciar os requisitos de forma eficaz. Essa colaboração entre a engenharia e os requisitos é fundamental para o sucesso de um projeto, garantindo que o produto atenda às necessidades dos stakeholders e alcance seus objetivos.

Assim como muitas outras gestões, a engenharia de requisitos também tem o seus e destaca (FREITAS, 2019) que eles podem variar dependendo de três principais aspectos: domínio, pessoas envolvidas e cultura e política organizacional. Esses fatores influenciam a forma como os processos de engenharia de requisitos são executados em diferentes projetos e organizações.

1. Domínio: O domínio refere-se ao campo específico em que o projeto está inserido, como saúde, finanças, transporte, entre outros. Cada domínio possui características e requisitos próprios, o que pode impactar os processos de engenharia de requisitos. Por exemplo, as necessidades e regulamentações em um projeto de saúde podem ser diferentes das de um projeto de finanças;
2. Pessoas envolvidas: As pessoas envolvidas no projeto desempenham um papel importante na engenharia de requisitos. Isso inclui stakeholders, como clientes, usuários finais, gerentes de projeto e desenvolvedores. Cada um desses indivíduos tem diferentes perspectivas, necessidades e expectativas em relação aos requisitos do projeto. Portanto, os processos de engenharia de requisitos devem levar em consideração a interação e a colaboração entre essas diferentes partes interessadas;
3. Cultura e política organizacional: A cultura e a política organizacional de uma empresa podem influenciar a forma como os processos de engenharia de requisitos são estabelecidos e executados. Algumas organizações possuem abordagens mais formais e estruturadas, enquanto outras adotam abordagens mais flexíveis e ágeis. Além disso, a cultura organizacional também pode afetar a comunicação, a colaboração e a tomada de decisões relacionadas aos requisitos do projeto.

Apesar das variações nos processos de engenharia de requisitos, existem atividades genéricas comuns a todos os projetos.

### Estudo de viabilidade

O estudo de viabilidade é uma etapa crucial no processo de engenharia de requisitos, que visa analisar se um projeto é viável em termos técnicos, econômicos, legais e operacionais. Ele é realizado no início do ciclo de vida de um projeto para determinar se a implementação do sistema proposto é factível e benéfica para a organização.

O objetivo principal do estudo de viabilidade é avaliar se o projeto é realista e se os benefícios esperados superam os custos, riscos e restrições envolvidos. Essa análise envolve a avaliação de diversos aspectos, como a adequação tecnológica, o impacto financeiro, a conformidade legal e regulatória, a disponibilidade de recursos e a aceitação dos stakeholders.

Na engenharia de requisitos, existem diferentes tipos de viabilidade que são considerados:

* Viabilidade técnica: avalia se a tecnologia necessária para implementar o sistema proposto está disponível ou pode ser desenvolvida dentro dos prazos e recursos definidos. Isso inclui considerar a infraestrutura existente, a capacidade técnica da equipe e a compatibilidade com os sistemas e equipamentos já em uso;
* Viabilidade econômica: analisa se o projeto é financeiramente viável e se os benefícios esperados justificam os custos envolvidos. Isso envolve a estimativa de investimentos necessários, os custos de desenvolvimento, manutenção e operação do sistema, assim como a projeção de retorno sobre o investimento;
* Viabilidade legal e regulatória: verifica se o projeto está em conformidade com as leis, regulamentos e normas aplicáveis. Isso inclui aspectos como proteção de dados, direitos autorais, propriedade intelectual e outras obrigações legais e éticas;
* Viabilidade operacional: analisa se o sistema proposto é factível em termos operacionais, considerando a disponibilidade de recursos humanos, infraestrutura física e capacidade organizacional. Também leva em conta a aceitação dos usuários finais e o impacto nas operações existentes.

No filme *Iron Man* (2008), temos um exemplo de projeto inviável tecnicamente. O personagem Howard Stark, pai de Tony Stark (o Homem de Ferro), menciona uma tecnologia inviável de ser desenvolvida na época. Durante uma gravação deixada para Tony (figura 3), Howard Stark menciona o "Elemento Stark" como uma descoberta que estava além do seu tempo e que ele não tinha conhecimento suficiente para desenvolver completamente.

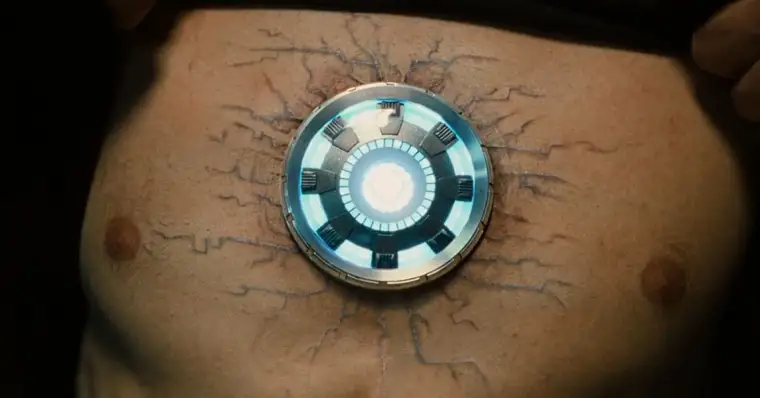
Figura 3 - Howard Stark, no filme Iron Man (2018), aparecendo em gravação para seu filho citando uma tecnologia inviável de ser construída em seu tempo devido a inviabilidade técnica



Fonte: Procolos Marvel, 2023[[2]](#footnote-2)

Essa menção ao "Elemento Stark" foi uma maneira de mostrar que Howard Stark estava trabalhando em uma tecnologia inovadora, mas que não tinha sido completamente desenvolvida ou compreendida. Isso cria um mistério em torno desse elemento e, posteriormente, no universo cinematográfico da Marvel, o "Elemento Stark" é revelado como sendo a base para a criação do reator *arc* (figura 4), que alimenta o traje do Homem de Ferro.

Figura 4 – Reator *arc:* um dispositivo inviável de ser construído na época do Howard Stark, alimentado por uma substância chamada paládio, que é capaz de gerar uma poderosa fonte de energia limpa e sustentar o traje do Homem de Ferro



Fonte: Legião dos Heróis, 2023?[[3]](#footnote-3)

Essa referência, de uma tecnologia inviável, no filme serve como algo tangível para que possamos entender a inviabilidade técnica.

### Elicitação de requisitos

A elicitação de requisitos é uma fase fundamental do processo de engenharia de requisitos, que envolve a coleta e o entendimento das necessidades, expectativas e restrições dos stakeholders (partes interessadas) em relação a um sistema ou produto a ser desenvolvido. É um processo interativo e colaborativo que visa identificar, analisar e documentar os requisitos de forma precisa e completa.

A elicitação de requisitos desempenha um papel crucial no sucesso de um projeto, uma vez que a compreensão correta e abrangente dos requisitos é essencial para o desenvolvimento de um sistema que atenda às expectativas dos usuários finais. Nessa etapa, é fundamental estabelecer uma comunicação clara e efetiva com os stakeholders, buscando compreender suas necessidades reais e as restrições do ambiente em que o sistema será implantado.

Durante o processo de elicitação de requisitos, é essencial descobrir e compreender o domínio da aplicação, os serviços requeridos e os vínculos e limitações relacionados ao sistema a ser desenvolvido. Vamos explorar cada um desses aspectos com mais detalhes:

* Domínio da aplicação: Compreender o domínio da aplicação significa obter conhecimento sobre o contexto em que o sistema será utilizado. Isso envolve identificar e entender o ambiente, os processos e as atividades existentes relacionadas ao sistema. Essa compreensão ajuda a estabelecer uma base sólida para definir os requisitos, garantindo que o sistema seja adequado e relevante para o domínio específico em questão;
* Serviços requeridos: Durante a elicitação de requisitos, é necessário identificar os serviços que o sistema deve fornecer aos seus usuários ou stakeholders. Esses serviços são funcionalidades, comportamentos ou capacidades específicas que o sistema deve oferecer para atender às necessidades e expectativas dos usuários. Através da identificação dos serviços requeridos, é possível definir os requisitos funcionais, ou seja, as ações que o sistema deve ser capaz de realizar;
* Vínculos e limitações: Os vínculos referem-se às interações e dependências do sistema com outros sistemas, componentes ou elementos do ambiente em que está inserido. Durante a elicitação de requisitos, é importante identificar e compreender esses vínculos para garantir que o sistema seja projetado e implementado de maneira consistente e integrada ao contexto em que será utilizado.

As limitações, por sua vez, se referem a restrições, restrições orçamentárias, restrições de tempo, restrições tecnológicas, restrições regulatórias, entre outras. Essas limitações devem ser levadas em consideração ao definir os requisitos do sistema, uma vez que podem influenciar o escopo, as funcionalidades e as restrições do projeto.

A descoberta e a compreensão desses aspectos durante a elicitação de requisitos são fundamentais para definir requisitos claros, completos e consistentes. Essas informações ajudam a estabelecer uma base sólida para o processo de desenvolvimento do sistema, permitindo que os engenheiros de requisitos e as partes interessadas tenham uma compreensão compartilhada do que é necessário para atender às expectativas e requisitos do projeto.

É importante ressaltar que a elicitação de requisitos é um processo contínuo e iterativo. À medida que os requisitos são coletados, eles precisam ser validados, refinados e priorizados, garantindo que sejam claros, consistentes, completos e testáveis. Além disso, é essencial estabelecer um mecanismo de comunicação efetivo com os stakeholders ao longo de todo o processo de desenvolvimento do sistema para lidar com mudanças e novos requisitos que possam surgir.

A elicitação de requisitos é um desafio complexo, pois envolve diferentes perspectivas e expectativas dos stakeholders. Alguns dos problemas típicos incluem:

* Falta de clareza nas necessidades dos stakeholders: Os stakeholders podem ter dificuldade em expressar de forma clara e precisa suas necessidades e expectativas em relação ao sistema. Isso pode levar a ambiguidades, requisitos mal definidos ou incompletos. É importante realizar uma comunicação efetiva e aberta para garantir que todas as partes envolvidas compreendam e expressem adequadamente suas necessidades;
* Conflito de terminologia e linguagem: Diferentes stakeholders podem ter diferentes backgrounds e experiências, o que pode levar a conflitos de terminologia e linguagem. É comum que os termos técnicos sejam usados pelos especialistas de tecnologia, enquanto os stakeholders de negócios podem ter uma compreensão diferente. Essa discrepância pode dificultar a comunicação e a compreensão mútua, exigindo esforços para estabelecer uma linguagem comum e definir termos de forma clara;
* Conflito de requisitos: Às vezes, os requisitos podem entrar em conflito entre si, com um requisito impedindo a realização de outro. Isso pode ser resultado de diferentes perspectivas e prioridades dos stakeholders. É necessário gerenciar esses conflitos, analisar as compensações necessárias e buscar soluções que atendam às necessidades e expectativas da maioria dos stakeholders;
* Impactos organizacionais e políticos: A introdução de um novo sistema pode ter impactos significativos nas operações, estrutura organizacional e poder político dentro de uma organização. Pode haver resistência a mudanças, disputas de poder ou restrições políticas que afetam a elicitação de requisitos e a definição de soluções. É importante ter em mente esses aspectos e envolver todas as partes interessadas relevantes para mitigar esses desafios;
* Processo dinâmico: O processo de elicitação de requisitos é dinâmico, sujeito a mudanças ao longo do tempo. Requisitos podem ser eliminados, alterados ou novos requisitos podem surgir à medida que o entendimento do sistema evolui ou as necessidades dos stakeholders mudam. Além disso, os stakeholders podem mudar ao longo do tempo, o que requer uma gestão contínua da elicitação de requisitos para manter a validade e relevância dos requisitos capturados;
* Alteração do ambiente: O ambiente em que o sistema será implantado pode mudar ao longo do tempo. Novas regulamentações, tecnologias emergentes, mudanças econômicas e outras variáveis externas podem impactar os requisitos do sistema. É necessário monitorar e adaptar-se a essas mudanças para garantir que o sistema seja viável e atenda às necessidades em constante evolução.

Ao enfrentar esses problemas, é importante adotar abordagens de colaboração, comunicação aberta e contínua com os stakeholders, além de utilizar técnicas adequadas de elicitação de requisitos para garantir um processo efetivo e um entendimento compartilhado das necessidades e requisitos do sistema.

Uma abordagem cuidadosa e colaborativa, juntamente com o uso das técnicas apropriadas, pode ajudar a garantir que os requisitos sejam compreendidos e capturados de maneira adequada, fornecendo uma base sólida para o sucesso do projeto.

Existem várias técnicas e abordagens utilizadas na elicitação de requisitos, e a escolha da melhor técnica depende do contexto do projeto e das características dos stakeholders envolvidos.

#### Questionários

Questionários são uma das técnicas comumente utilizadas na elicitação de requisitos. Eles envolvem a criação de um conjunto estruturado de perguntas para coletar informações dos stakeholders sobre suas necessidades, expectativas e requisitos relacionados ao sistema a ser desenvolvido. Os questionários podem ser distribuídos eletronicamente, por e-mail ou por meio de plataformas online, ou podem ser preenchidos em formato impresso.

A utilização de questionários na elicitação de requisitos apresenta algumas vantagens:

1. Escala e abrangência: Os questionários permitem coletar informações de um grande número de stakeholders de forma eficiente. Eles podem ser distribuídos para um público amplo, alcançando um maior número de pessoas e obtendo uma visão abrangente das necessidades e requisitos;
2. Anonimato e privacidade: Os questionários podem ser respondidos anonimamente pelos stakeholders, o que pode encorajar a expressão franca de opiniões e necessidades. Isso é especialmente útil quando há preocupações sobre possíveis retaliações ou constrangimentos associados à divulgação de informações;
3. Padronização e consistência: Os questionários permitem a padronização das perguntas e respostas, garantindo que todos os stakeholders sejam apresentados às mesmas questões. Isso ajuda a obter respostas consistentes e comparáveis, facilitando a análise posterior dos dados coletados;
4. Menor influência de viés pessoal: Ao responder um questionário, os stakeholders têm tempo para refletir sobre suas respostas e fornecer informações objetivas. Isso ajuda a minimizar a influência de fatores pessoais, emoções ou pressões externas que podem ocorrer durante entrevistas ou reuniões presenciais.

No entanto, é importante estar ciente das limitações dos questionários na elicitação de requisitos:

1. Falta de interação direta: Os questionários não permitem uma interação direta e imediata entre os stakeholders e os analistas de requisitos. Isso pode limitar a capacidade de esclarecer dúvidas, obter informações adicionais ou explorar questões mais profundamente.
2. Respostas limitadas: Os questionários geralmente fornecem opções de resposta pré-definidas ou exigem respostas curtas. Isso pode restringir a capacidade dos stakeholders de expressar completamente suas necessidades ou fornecer informações contextuais relevantes.
3. Possibilidade de respostas incompletas ou imprecisas: Sem a presença de um facilitador para esclarecer dúvidas ou fornecer orientações, os stakeholders podem não compreender completamente as perguntas ou podem fornecer respostas incompletas ou imprecisas. Isso pode resultar na obtenção de informações limitadas ou ambíguas.

Para maximizar a eficácia dos questionários na elicitação de requisitos, é recomendado seguir as boas práticas, como:

* Elaborar perguntas claras, concisas e não ambíguas;
* Fornece instruções claras sobre como preencher o questionário;
* Oferecer opções de resposta adequadas, como escalas *Likert*, múltipla escolha ou campos de preenchimento livre;
* Incluir espaço para comentários ou informações adicionais, caso os stakeholders queiram fornecer mais detalhes;
* Realizar uma revisão e teste prévio.

#### Entrevistas

As entrevistas são uma das técnicas mais populares e eficazes na elicitação de requisitos. Elas envolvem uma interação direta entre os analistas de requisitos e os stakeholders, permitindo obter informações detalhadas sobre suas necessidades, expectativas e requisitos relacionados ao sistema em desenvolvimento.

As entrevistas podem ser conduzidas de maneira presencial, por telefone ou por meio de videoconferências, dependendo da disponibilidade e localização dos stakeholders. Durante as entrevistas, os analistas de requisitos fazem perguntas abertas e estruturadas para explorar o conhecimento e a perspectiva dos stakeholders em relação ao sistema. A seguir, na figura 5, uma ilustração de entrevista com cliente:

Figura 5 - Ilustração de entrevista com cliente para elicitação de requisitos



Fonte: Zendesk, 2020[[4]](#footnote-4)

Existem várias vantagens associadas ao uso de entrevistas na elicitação de requisitos:

1. Compreensão aprofundada: As entrevistas permitem uma compreensão mais aprofundada das necessidades e requisitos dos stakeholders. Os analistas de requisitos podem fazer perguntas adicionais, esclarecer dúvidas, explorar casos de uso e cenários específicos, o que ajuda a obter informações detalhadas e contextuais;
2. Interação direta: A interação direta entre os analistas de requisitos e os stakeholders facilita o estabelecimento de um relacionamento de confiança. Isso encoraja os stakeholders a compartilharem informações mais abertamente e a expressarem suas opiniões, preocupações e expectativas de maneira mais completa;
3. Feedback imediato: Durante as entrevistas, os analistas de requisitos podem receber feedback imediato dos stakeholders. Isso permite a verificação de entendimento, esclarecimento de pontos obscuros e ajustes no processo de elicitação conforme necessário;
4. Flexibilidade: As entrevistas podem ser adaptadas de acordo com as necessidades e características dos stakeholders. Os analistas de requisitos podem personalizar as perguntas e a abordagem de acordo com a área de especialização do stakeholder, seu nível de conhecimento técnico e seu papel no projeto.

No entanto, também existem algumas considerações importantes ao utilizar entrevistas na elicitação de requisitos:

1. Viés do entrevistador: O entrevistador pode introduzir viés nas perguntas ou nas interpretações das respostas. É importante que os analistas de requisitos estejam cientes disso e sejam neutros ao conduzir as entrevistas;
2. Disponibilidade dos stakeholders: Agendar entrevistas com os stakeholders pode ser desafiador, especialmente quando eles estão localizados em diferentes locais ou têm agendas ocupadas. É necessário planejar com antecedência e coordenar as entrevistas de acordo com a disponibilidade de todos os envolvidos;
3. Dependência de habilidades de comunicação: O sucesso das entrevistas depende das habilidades de comunicação dos analistas de requisitos. Eles devem ser capazes de fazer perguntas relevantes, ouvir ativamente, capturar informações-chave e sintetizar as respostas dos stakeholders;
4. Limitação de amostragem: As entrevistas têm uma amostragem limitada, o que significa que nem todos os stakeholders podem ser entrevistados. É importante garantir que uma variedade de stakeholders relevantes seja incluída nas entrevistas para obter uma perspectiva representativa.

Em resumo, as entrevistas são uma técnica valiosa na elicitação de requisitos, pois permitem uma compreensão aprofundada das necessidades e requisitos dos stakeholders.

#### Dinâmica de grupo

A dinâmica de grupo, mais especificamente o brainstorming, é uma técnica comumente utilizada na elicitação de requisitos. Essa técnica envolve a reunião de um grupo de pessoas (figura 6) incluindo analistas de requisitos e stakeholders relevantes, para gerar ideias, explorar possibilidades e obter contribuições colaborativas para a definição dos requisitos do sistema.

Figura 6 - Ilustração de uma reunião de brainstorming



Fonte: Volusion, 2016[[5]](#footnote-5)

Durante uma sessão de brainstorming, os participantes são encorajados a expressar suas ideias de forma livre e espontânea, sem restrições ou julgamentos. O objetivo é fomentar a criatividade e a geração de novas perspectivas sobre os requisitos do sistema. Algumas diretrizes comuns para o brainstorming incluem:

1. Definir um objetivo claro: Antes da sessão de brainstorming, é importante estabelecer um objetivo específico para direcionar a discussão. Isso ajuda os participantes a focarem suas ideias e contribuições na direção correta;
2. Criar um ambiente acolhedor: É fundamental estabelecer um ambiente no qual todos os participantes se sintam à vontade para compartilhar suas ideias sem medo de críticas ou julgamentos. A abertura e o respeito mútuo são essenciais para estimular a participação ativa de todos;
3. Encorajar a diversidade de ideias: Durante a sessão de brainstorming, é importante incentivar a diversidade de ideias e perspectivas. Isso pode ser alcançado encorajando os participantes a pensar fora da caixa, explorar soluções não convencionais e considerar diferentes abordagens para atender aos requisitos;
4. Promover a colaboração: O brainstorming é uma atividade colaborativa, na qual os participantes constroem e ampliam as ideias uns dos outros. É importante criar oportunidades para discussões e interações entre os participantes, permitindo que eles se inspirem mutuamente e desenvolvam conceitos mais refinados;
5. Capturar todas as ideias: Durante a sessão de brainstorming, é fundamental registrar todas as ideias geradas, independentemente de sua viabilidade ou relevância imediata. Essas ideias podem servir como base para discussões futuras e podem conter insights valiosos para o desenvolvimento do sistema;
6. Estimular a criatividade: É benéfico utilizar técnicas que estimulem a criatividade durante o brainstorming. Isso pode incluir o uso de métodos visuais, como quadros brancos ou murais, a aplicação de técnicas de associação livre, a utilização de estímulos visuais ou até mesmo a exploração de analogias ou metáforas.

Após a sessão de brainstorming, as ideias geradas podem ser analisadas, refinadas e classificadas por sua viabilidade, relevância e impacto. É possível utilizar outras técnicas de análise de requisitos, como a priorização ou a categorização, para identificar os requisitos mais importantes e valiosos para o projeto.

O brainstorming como técnica de elicitação de requisitos permite uma participação ativa dos stakeholders, promove a criatividade e a inovação, e pode resultar em uma compreensão mais rica e abrangente das necessidades e requisitos do sistema a ser desenvolvido.

Existe uma variação dessa técnica chamada brainwriting. Sim, o brainwriting é uma variação da técnica de brainstorming. Nessa abordagem, ao invés de compartilhar ideias oralmente, os participantes as registram por escrito de forma simultânea e anônima (figura 7).

O processo do brainwriting geralmente ocorre da seguinte maneira:

1. Preparação: Reúna um grupo de participantes e defina o objetivo da sessão de brainwriting. Certifique-se de que todos tenham acesso a papel e caneta;
2. Regras de anonimato: Estabeleça que as ideias devem ser registradas anonimamente, sem a identificação do autor. Isso permite que os participantes se sintam mais à vontade para contribuir e evita a influência de hierarquias ou preconceitos;
3. Geração de ideias: Cada participante recebe uma folha de papel em branco e é instruído a escrever uma ideia relacionada ao objetivo definido. Após alguns minutos, as folhas são dobradas e passadas para o participante ao lado;
4. Expansão de ideias: O próximo participante recebe a folha dobrada, lê a ideia anterior e é incentivado a expandi-la, adicionar detalhes ou gerar uma nova ideia com base naquela. Novamente, após alguns minutos, as folhas são dobradas e passadas para a próxima pessoa;
5. Iteração do processo: O processo de expansão de ideias continua até que todas as folhas tenham circulado entre todos os participantes. Cada participante tem a oportunidade de ler as ideias anteriores e contribuir com novas perspectivas;
6. Discussão e análise: Após a conclusão do processo de brainwriting, as folhas são abertas e as ideias são compartilhadas em grupo. As ideias podem ser discutidas, agrupadas, refinadas e avaliadas em termos de relevância e viabilidade.

Figura 7 - Sessão de brainwriting conduzida por Murilo Gun, fundador da *Keep Learning School*



Fonte: Keep Learning School, 2021[[6]](#footnote-6)

O brainwriting oferece algumas vantagens em relação ao brainstorming tradicional:

* Maior participação equitativa: O anonimato no brainwriting permite que todos os participantes contribuam de forma mais igualitária, independentemente de sua posição hierárquica ou personalidade mais introvertida. Isso evita a influência de ideias dominantes e estimula uma gama mais ampla de perspectivas;
* Menos interrupções e pressões sociais: Como as ideias são registradas por escrito, os participantes têm tempo para refletir e desenvolver suas contribuições sem interrupções ou pressões sociais. Isso promove uma maior diversidade de pensamento e evita a tendência de conformidade ou polarização;
* Geração de mais ideias: O processo de iteração do brainwriting permite que as ideias sejam continuamente expandidas e refinadas. Isso resulta em um maior volume de ideias geradas em comparação com o brainstorming tradicional.

No entanto, assim como o brainstorming, o brainwriting também possui algumas considerações:

* Menos interação direta: Como as ideias são registradas por escrito, pode haver menos interação verbal e menos oportunidades para discussões imediatas e clarificação de ideias;
* Limitação de tempo: É importante estabelecer limites de tempo adequados para cada fase do processo de brainwriting, a fim de garantir que ele seja conduzido de maneira eficiente e produtiva;
* Necessidade de síntese e análise posterior: Após a sessão de brainwriting, é necessário dedicar tempo para revisar, sintetizar e analisar as ideias geradas. Isso pode envolver a categorização, priorização ou combinação de ideias semelhantes.

O brainwriting é uma técnica valiosa para a elicitação de requisitos, especialmente em situações em que se busca uma maior participação equitativa e uma geração ampla de ideias. É uma abordagem eficaz para explorar diferentes perspectivas e estimular a criatividade dentro de um grupo de trabalho.

#### Prototipação

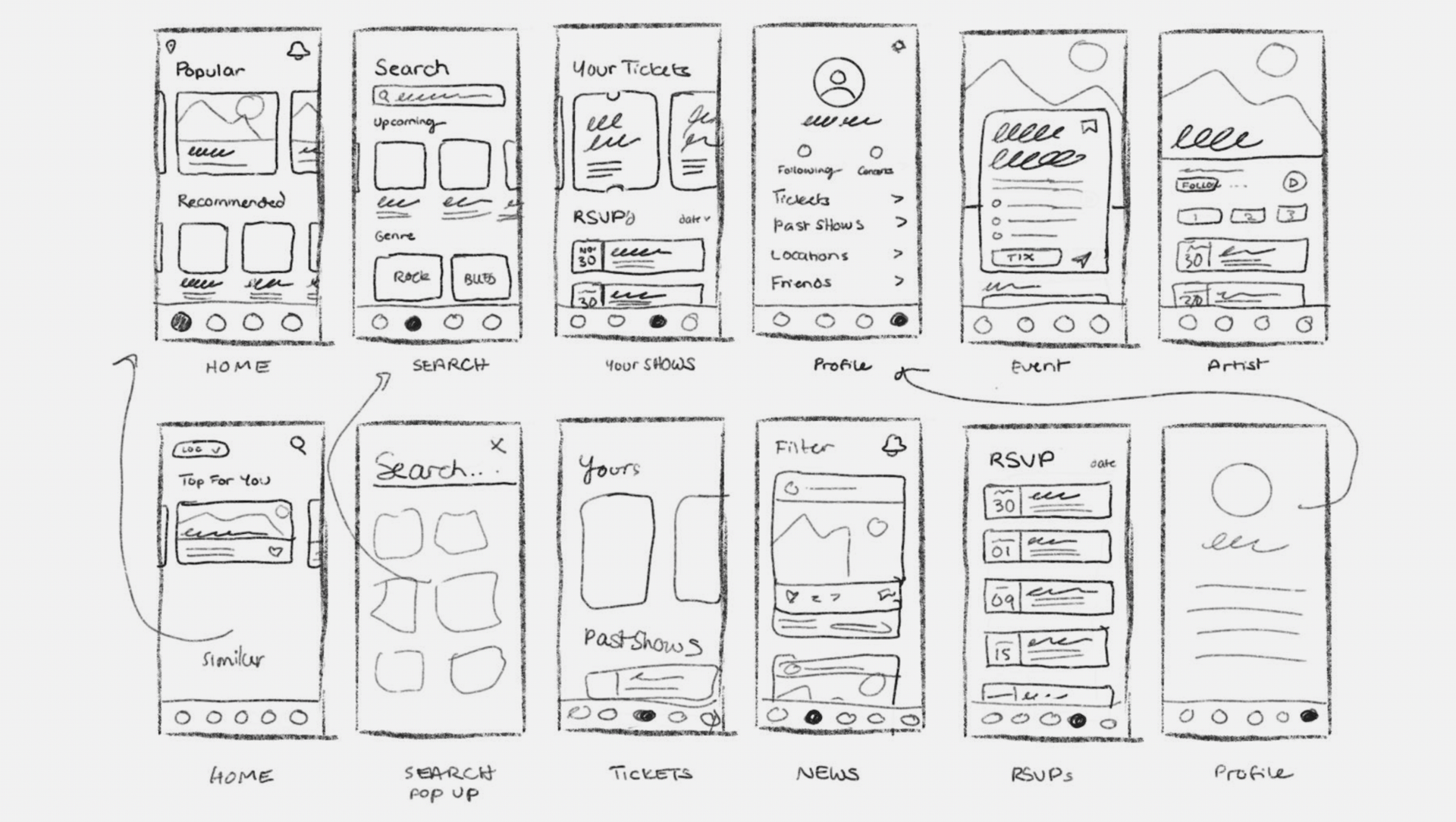
A prototipação é uma técnica amplamente utilizada no processo de elicitação de requisitos de software. Consiste em criar representações simplificadas e interativas do sistema em desenvolvimento, permitindo aos stakeholders visualizarem e experimentarem funcionalidades e características antes da implementação final.

A prototipação ajuda a facilitar a comunicação entre os desenvolvedores e os usuários finais, além de fornecer uma base tangível para discussões e validações dos requisitos. Ao criar protótipos, os requisitos são mais facilmente compreendidos e avaliados pelos stakeholders, permitindo identificar problemas, melhorias e refinamentos necessários no sistema.

Existem diferentes abordagens de prototipação, como a prototipação descartável, a evolutiva e a de alto nível. A prototipação descartável envolve a criação rápida de protótipos que não são destinados a se tornarem parte do sistema final, sendo usados apenas para fins de exploração e entendimento dos requisitos. Já a prototipação evolutiva envolve a construção de protótipos iterativos que são gradualmente aprimorados e refinados, com base no feedback dos stakeholders. Esses protótipos evoluem ao longo do processo de desenvolvimento até se tornarem a versão final do sistema.

Como exemplo de prototipação descartável e evolutiva, dependendo do contexto e dos objetivos do projeto, existem os *wireframes*. Eles são uma forma comum de prototipação usada no processo de elicitação de requisitos. Eles são representações visuais básicas (figura 8) e esquemáticas das interfaces do usuário, geralmente sem cores ou detalhes visuais complexos.

Figura 8 - Exemplo de wireframes



Fonte: Sketch, 2022[[7]](#footnote-7)

Os wireframes são criados para fornecer uma estrutura visual do sistema, mostrando a disposição dos elementos de interface, a hierarquia das informações e os fluxos de interação. Eles se concentram principalmente na estrutura e na organização do conteúdo, ignorando detalhes visuais e estilísticos.

Os wireframes são úteis durante a fase inicial de elicitação de requisitos, na prototipação descartável, pois permitem que os stakeholders visualizem e compreendam a disposição geral das informações e a interação entre os diferentes elementos da interface. Eles podem ser usados para capturar requisitos de usabilidade, identificar problemas de fluxo de trabalho e realizar testes de usabilidade preliminares.

Ao criar wireframes, é importante envolver os stakeholders relevantes, como usuários finais e designers de interface, para obter feedback e validação. Os wireframes podem ser feitos de forma manual, por meio de esboços em papel ou quadros brancos, ou por meio de ferramentas especializadas de design de interfaces, que permitem criar wireframes digitais interativos.

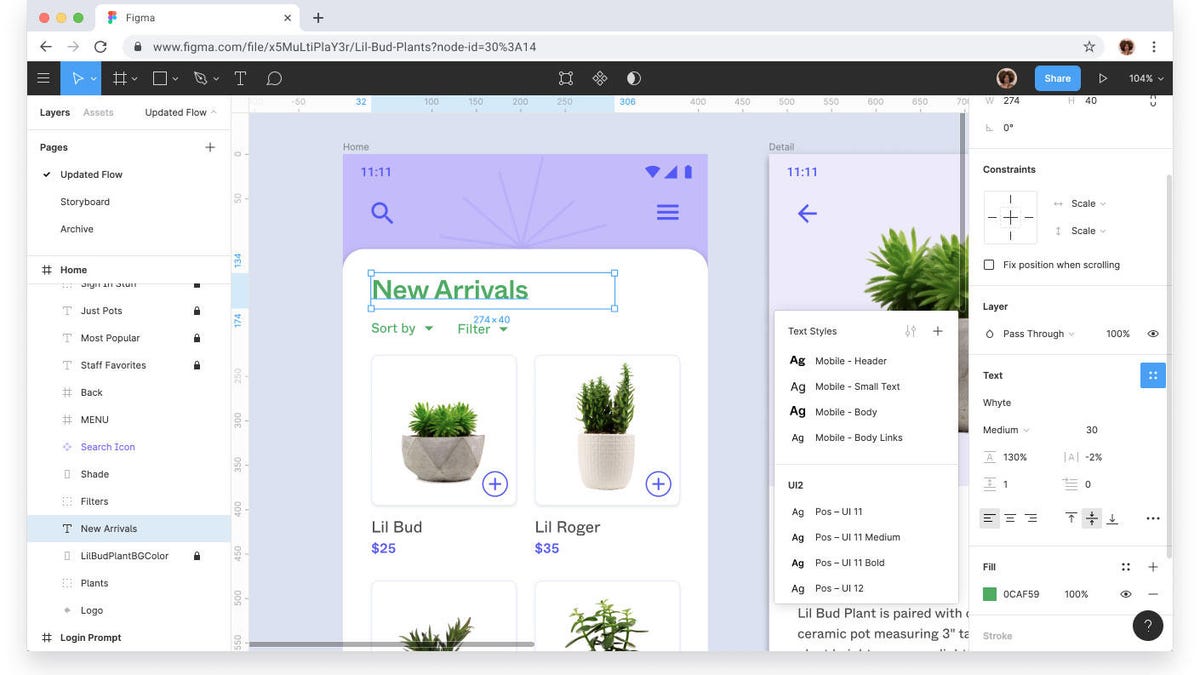
Os wireframes podem ser refinados e evoluídos ao longo do processo de elicitação de requisitos, à medida que mais detalhes e informações são obtidos. Eles podem servir como base para a criação de protótipos mais detalhados e interativos, à medida que o sistema é desenvolvido.

A prototipação de alto nível é focada na representação visual e interativa das principais funcionalidades e fluxos do sistema, sem se aprofundar em detalhes técnicos. Essa abordagem permite uma compreensão geral do sistema e facilita a validação das ideias e conceitos-chave.

Existem algumas ferramentas no mercado para desenvolver esse tipo de protótipo. O Figma e o Adobe XD são exemplos populares.

O Figma (figura 9) é uma ferramenta baseada em nuvem que permite a criação colaborativa de designs e protótipos. Ele oferece recursos abrangentes de design de interface, como criação de wireframes, criação de layouts responsivos, criação de elementos de interface personalizados, além de permitir a criação de protótipos interativos com transições e animações.

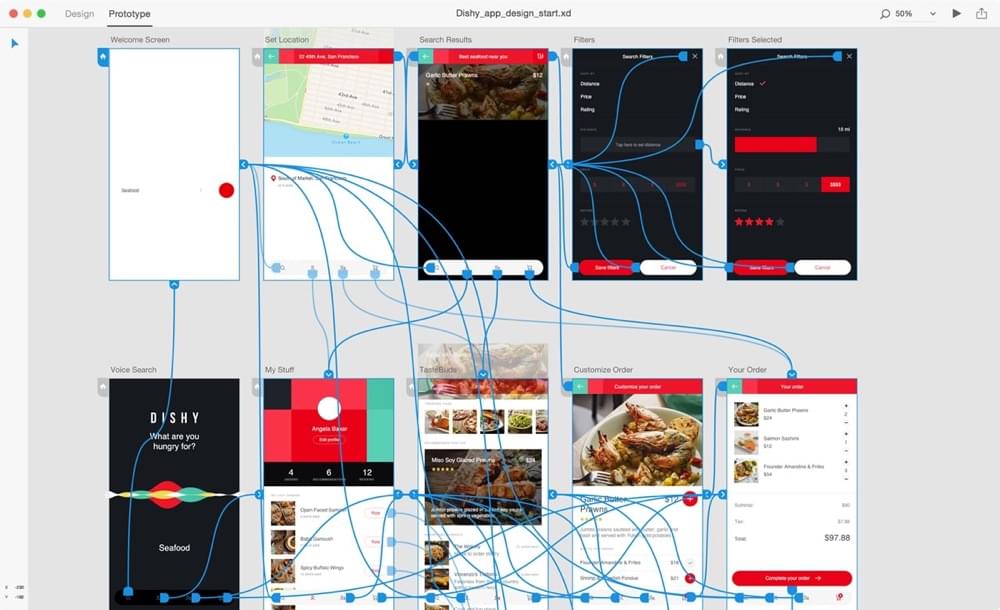
Figura 9 - Figma como ferramenta para desenvolver protótipos de alto nível



Fonte: Cnet, 2019[[8]](#footnote-8)

O Adobe XD (figura 10) é uma ferramenta da Adobe desenvolvida especificamente para design de interfaces e prototipação. Ele também oferece recursos para criação de wireframes, design de interfaces responsivas, criação de componentes reutilizáveis e criação de protótipos interativos com recursos de transição e animação.

Figura 10 - Adobe XD como ferramenta para desenvolver protótipos de alto nível



Fonte: Sitepoint, 2017[[9]](#footnote-9)

Ao utilizar a prototipação como técnica de elicitação de requisitos, é importante considerar seus benefícios, como a facilitação da compreensão dos requisitos, a validação precoce das ideias, a identificação antecipada de problemas e a redução de retrabalho. No entanto, também é necessário ter em mente que os protótipos são simplificações do sistema e podem não capturar todos os aspectos e complexidades do sistema final.

#### Etnografia

A etnografia é uma técnica de elicitação de requisitos que se baseia na observação direta e imersão dos analistas de requisitos no ambiente em que o sistema será utilizado. Essa abordagem visa compreender profundamente o contexto, as atividades, as interações sociais e as práticas dos usuários, a fim de identificar necessidades e requisitos de forma mais precisa.

Ao utilizar a etnografia como técnica de elicitação de requisitos, os analistas de requisitos passam um tempo significativo no ambiente de trabalho dos usuários finais, observando suas atividades, interagindo com eles e documentando suas descobertas. Isso permite que os analistas obtenham uma compreensão holística do sistema em seu contexto real, capturando detalhes e nuances que podem não ser evidentes em outras abordagens de elicitação de requisitos.

Além da observação, os analistas de requisitos também podem realizar entrevistas informais com os usuários durante a etnografia, buscando esclarecimentos e insights adicionais sobre suas experiências e necessidades. Essa abordagem permite que os requisitos sejam derivados de maneira mais natural e contextualizada, levando em consideração os aspectos culturais, comportamentais e sociais dos usuários.

A etnografia pode ser especialmente útil em projetos que envolvem sistemas complexos ou em ambientes onde as atividades dos usuários são difíceis de descrever ou explicar verbalmente. Ela ajuda a identificar requisitos não articulados ou implícitos, que podem ser cruciais para o sucesso do sistema.

No entanto, é importante ressaltar que a etnografia pode ser um processo demorado e exigir uma imersão aprofundada no ambiente do usuário. Além disso, os analistas de requisitos devem ser cuidadosos ao interpretar suas observações e garantir que não haja viés ou interpretação inadequada.

### Análise de requisitos

A análise de requisitos é uma atividade fundamental no processo de engenharia de requisitos, que tem como objetivo compreender, modelar, organizar e refinar os requisitos coletados durante a elicitação. A análise de requisitos visa aprofundar a compreensão dos requisitos e sua interdependência, identificar possíveis inconsistências, ambiguidades ou lacunas e estabelecer uma base sólida para o desenvolvimento do sistema.

Durante a análise de requisitos, várias técnicas e abordagens podem ser aplicadas para auxiliar na compreensão e refinamento dos requisitos. Alguns aspectos importantes abordados nessa atividade incluem:

* Classificação e categorização de requisitos: Os requisitos coletados podem ser classificados e categorizados para melhor organização e análise. Isso envolve agrupar requisitos semelhantes com base em características comuns, como funcionalidades, restrições ou necessidades dos usuários. A classificação e categorização facilitam a análise posterior e o entendimento dos diferentes aspectos do sistema.
* Priorização de requisitos: A priorização de requisitos é uma etapa essencial na análise, em que os requisitos são avaliados e classificados de acordo com sua importância e valor para os stakeholders. Isso permite a tomada de decisões informadas sobre quais requisitos devem ser implementados primeiro ou receber maior atenção. A priorização é baseada em critérios como impacto nos objetivos do projeto, riscos associados e valor agregado aos usuários finais.
* Análise de consistência e conflitos: Durante a análise de requisitos, é importante identificar e tratar inconsistências, conflitos ou ambiguidades entre os requisitos. Isso é feito por meio da revisão minuciosa dos requisitos, comparando-os e verificando se são mutuamente consistentes e compatíveis. Caso ocorram conflitos, é necessário buscar resoluções adequadas em colaboração com os stakeholders, buscando encontrar um equilíbrio e uma solução que atenda às necessidades de todos.
* Modelagem de requisitos: A modelagem de requisitos é uma técnica utilizada para representar os requisitos de forma visual e estruturada. Isso pode ser feito por meio de diagramas, como diagramas de caso de uso, diagramas de sequência, diagramas de atividades, entre outros. A modelagem ajuda a capturar e comunicar os requisitos de forma mais clara e compreensível, facilitando o entendimento tanto para os desenvolvedores quanto para os stakeholders.
* Refinamento de requisitos: Durante a análise, é comum que os requisitos sejam refinados, ou seja, detalhados e especificados de forma mais precisa. Isso pode envolver a decomposição de requisitos em requisitos mais detalhados, a definição de critérios de aceitação, a identificação de restrições ou a adição de informações adicionais para tornar os requisitos mais completos e compreensíveis.

A análise de requisitos é uma atividade iterativa e contínua, que ocorre ao longo do ciclo de vida do projeto. À medida que os requisitos são analisados e refinados, pode ser necessário retornar à fase de elicitação para obter mais informações ou esclarecimentos. Esse processo iterativo ajuda a garantir a qualidade e a adequação dos requisitos, além de facilitar a comunicação entre os envolvidos no projeto.

A modelagem gráfica é uma técnica bem importante e será vista com mais detalhe na seção 3.4.

Jeff Sutherland, um dos criadores da metodologia ágil Scrum, destaca a importância da priorização de requisitos no processo de desenvolvimento de software. A priorização de requisitos é fundamental para garantir que os recursos limitados, como tempo e orçamento, sejam alocados da maneira mais eficiente e eficaz possível.

Ao priorizar requisitos, as equipes de desenvolvimento e os stakeholders precisam identificar quais requisitos têm maior valor e impacto no sucesso do projeto. Isso envolve entender as necessidades dos usuários, os objetivos do negócio e os critérios de sucesso do projeto.

### Validação

A validação de requisitos é um processo importante durante o desenvolvimento de um projeto, pois visa garantir que os requisitos definidos sejam corretos, completos e atendam às necessidades e expectativas dos stakeholders. A validação de requisitos envolve a revisão, verificação e confirmação dos requisitos antes que eles sejam implementados.

Erros de requisitos apresentam alto custo; correção de erros pós entrega podem apresentar um custo 100 vezes superior ao de sua correção na fase de validação (FREITAS, 2022), o que ressalta a importância do processo de validação.

Quando erros de requisitos são identificados após o início do desenvolvimento, pode ser necessário realizar retrabalho significativo para corrigir ou adaptar o sistema. Isso consome recursos adicionais, como tempo e esforço da equipe de desenvolvimento, o que pode impactar o cronograma e o orçamento do projeto.

Também, podem levar a atrasos significativos no cronograma do projeto. Se os requisitos não forem corretamente compreendidos e documentados, é mais provável que ocorram problemas de comunicação e entendimento entre os stakeholders e a equipe de desenvolvimento. Isso pode resultar em atrasos na entrega do sistema finalizado.

Assim como descrito na introdução, erros podem impactar na satisfação do cliente. Se os requisitos não atenderem às expectativas e necessidades do cliente, isso irá gerar reclamações, insatisfação e, em última análise, à perda de negócios ou oportunidades futuras.

Se os erros de requisitos não forem identificados durante o desenvolvimento e forem introduzidos no sistema final, os custos de correção pós-implementação podem ser substanciais. Além do retrabalho necessário, pode ser necessário lidar com problemas de funcionalidade, desempenho ou segurança que surgiram devido aos erros de requisitos.

Erros terão um efeito cascata nos requisitos subsequentes, afetando o design, a implementação e o teste do sistema. Isso pode criar uma cadeia de erros e aumentar ainda mais os custos associados à sua correção.

Quando se trata de validar requisitos, é importante considerar várias características-chave dos requisitos. Aqui estão algumas delas:

1. Validade: A validade refere-se à precisão e pertinência dos requisitos em relação ao sistema que está sendo desenvolvido. Os requisitos devem estar alinhados com as necessidades e expectativas dos stakeholders e devem representar corretamente as funcionalidades e características que o sistema deve ter. Durante a validação, é necessário verificar se os requisitos são válidos e se realmente atendem aos objetivos do projeto;
2. Completeza: A completeza dos requisitos diz respeito à extensão em que todos os aspectos necessários do sistema estão devidamente especificados nos requisitos. Os requisitos devem cobrir todas as funcionalidades principais e requisitos não funcionais importantes do sistema. Durante a validação, é necessário garantir que nenhum requisito essencial tenha sido omitido e que todas as necessidades relevantes tenham sido abordadas;
3. Consistência: A consistência refere-se à ausência de contradições ou conflitos entre os requisitos. Durante a validação, é importante identificar e resolver qualquer inconsistência entre os requisitos para garantir que eles sejam compatíveis e que não haja informações conflitantes que possam levar a interpretações diferentes;
4. Realismo: O realismo dos requisitos diz respeito à viabilidade técnica, econômica e operacional dos requisitos estabelecidos. Durante a validação, é necessário avaliar se os requisitos são realistas e se podem ser implementados dentro dos recursos disponíveis e das restrições do projeto. É importante considerar fatores como viabilidade tecnológica, custo, recursos humanos e prazos;
5. Rastreabilidade: A rastreabilidade dos requisitos é a capacidade de acompanhar e documentar a origem, as alterações e as relações entre os requisitos. Durante a validação, é importante verificar se cada requisito está adequadamente rastreado, ou seja, se é possível identificar sua origem e rastrear qualquer alteração ou evolução ao longo do tempo. A rastreabilidade dos requisitos auxilia na gestão de mudanças e no entendimento do impacto de alterações em requisitos posteriores.

Essas características são fundamentais para garantir que os requisitos sejam de alta qualidade e atendam às necessidades do projeto. Durante o processo de validação, é necessário revisar, analisar e verificar cada requisito em relação a essas características, a fim de garantir que sejam corretos, completos, consistentes, realistas e rastreáveis.

Existem várias técnicas de validação de requisitos que podem ser utilizadas para garantir a qualidade e a adequação dos requisitos. Segue brevemente alguma delas:

* Revisão de requisitos: A revisão é uma técnica de validação que envolve a análise detalhada dos requisitos por meio de revisões formais ou informais. Pode envolver a participação de especialistas, partes interessadas e membros da equipe de projeto. Durante a revisão, os requisitos são analisados quanto à validade, completeza, consistência, realismo e rastreabilidade. As revisões podem ser conduzidas por meio de reuniões presenciais ou virtuais, onde os participantes podem fornecer feedback, identificar problemas e sugerir melhorias nos requisitos;
* Prototipação: A prototipação é uma técnica em que são criados protótipos do sistema ou de partes específicas do sistema para validar os requisitos. Os protótipos são versões simplificadas ou simuladas do sistema que permitem aos stakeholders visualizarem e interagirem com as funcionalidades propostas. Através da prototipação, é possível identificar requisitos ausentes, validar a usabilidade, obter feedback dos usuários e verificar se os requisitos atendem às necessidades e expectativas;
* Casos de teste: Os casos de teste são desenvolvidos para validar os requisitos por meio de testes específicos. Cada caso de teste representa uma situação ou cenário em que o sistema é testado em relação aos requisitos estabelecidos. Os casos de teste podem abordar tanto os requisitos funcionais quanto os não funcionais. A execução dos casos de teste permite verificar se o sistema atende aos requisitos e identificar falhas ou discrepâncias. Os resultados dos testes podem fornecer informações valiosas para ajustes e melhorias nos requisitos.

## Modelagem gráfica

A modelagem gráfica na engenharia de requisitos é uma abordagem que utiliza representações visuais, como diagramas e gráficos, para descrever e representar os requisitos de um sistema de forma clara e concisa. Ela oferece uma maneira visualmente compreensível de comunicar as informações relacionadas aos requisitos entre as partes interessadas envolvidas no desenvolvimento de um sistema ou software.

A importância de modelar graficamente na engenharia de requisitos pode ser explicada por diversos motivos:

* Comunicação efetiva: A modelagem gráfica permite uma comunicação mais eficaz entre as partes interessadas, independentemente de seu conhecimento técnico. Diagramas visuais são mais intuitivos e fáceis de compreender do que descrições textuais extensas, facilitando a comunicação e o compartilhamento de ideias entre os membros da equipe de desenvolvimento, clientes e usuários finais;
* Visualização do sistema: Diagramas gráficos oferecem uma representação visual do sistema, permitindo que os envolvidos entendam sua estrutura, comportamento e interações de forma mais clara. Eles proporcionam uma visão holística do sistema, identificando seus componentes, suas relações e como eles funcionam em conjunto;
* Identificação de requisitos faltantes ou conflitantes: Ao modelar graficamente os requisitos, é possível identificar lacunas e inconsistências no sistema. Os diagramas podem ajudar a identificar requisitos que foram omitidos ou mal-entendidos, bem como possíveis conflitos entre diferentes partes do sistema. Isso permite que os problemas sejam detectados precocemente e corrigidos antes da implementação, economizando tempo e recursos;
* Análise e validação dos requisitos: A modelagem gráfica possibilita a análise e validação dos requisitos de forma mais efetiva. Os diagramas podem ser usados para verificar a consistência, a completude e a corretude dos requisitos, além de permitir a identificação de possíveis melhorias ou refinamentos. Através da análise visual, é possível identificar possíveis erros ou ambiguidades nos requisitos, garantindo uma especificação mais precisa e confiável;
* Documentação do sistema: Os diagramas gráficos servem como uma forma de documentação do sistema, permitindo que as informações relacionadas aos requisitos sejam registradas de maneira organizada e acessível. Eles fornecem uma representação visual que pode ser referenciada ao longo do ciclo de vida do sistema, facilitando a compreensão e a manutenção do sistema ao longo do tempo.

Os diagramas *Unified Modeling Language* (UML) e Caso de Uso são exemplos de técnicas de modelagem gráfica amplamente utilizadas na engenharia de requisitos. O UML é uma linguagem padronizada para modelagem de sistemas que inclui vários tipos de diagramas, como diagramas de classe, diagramas de sequência, diagramas de atividade, entre outros. Esses diagramas são usados para representar diferentes aspectos do sistema, desde sua estrutura até seu comportamento dinâmico.

Os Casos de Uso são uma técnica específica dentro do UML, que envolve a identificação e descrição dos atores (usuários ou sistemas externos) e suas interações com o sistema. Os diagramas de Caso de Uso representam essas interações, mostrando os cenários de uso do sistema e as funcionalidades que ele oferece aos seus usuários.

Ambas as técnicas são valiosas para modelar graficamente os requisitos de um sistema, pois oferecem uma representação visual que pode ser facilmente compreendida pelas partes interessadas. Além disso, elas permitem uma análise mais detalhada do sistema, ajudando a identificar requisitos, comportamentos e interações importantes que devem ser considerados durante o processo de desenvolvimento.

Antes do surgimento da UML, havia uma falta de padronização na representação visual do comportamento dos sistemas de software. Diferentes engenheiros de software em todo o mundo usavam seus próprios padrões e diagramas, o que tornava difícil a comunicação efetiva e a compreensão mútua dos requisitos e comportamentos do sistema.

O UML foi introduzido na década de 1990 por um grupo de especialistas em engenharia de software, liderado por Grady Booch, James Rumbaugh e Ivar Jacobson. O objetivo do UML era fornecer uma linguagem comum e um conjunto de diagramas para representar visualmente os aspectos estruturais e comportamentais de um sistema.

Com o UML, os engenheiros de software têm uma linguagem comum para descrever, analisar, projetar e comunicar os requisitos e o comportamento dos sistemas. Os diagramas do UML, como diagramas de casos de uso, diagramas de sequência, diagramas de classes e diagramas de atividades, fornecem uma forma padronizada de representar visualmente diferentes aspectos do sistema.

A adoção do UML trouxe benefícios significativos para a comunidade de engenheiros de software. Agora, independentemente da localização geográfica ou da formação individual, os engenheiros podem utilizar os mesmos padrões de representação e se comunicar de forma mais eficiente. O UML também facilita a documentação e a compreensão dos requisitos do sistema, permitindo uma análise mais precisa e uma colaboração mais efetiva entre as equipes de desenvolvimento.

Antes de começar a modelar uma aplicação, é necessário coletar os requisitos do sistema. Numa sessão de brainstorming, por exemplo, ideias do que o sistema pode fazer e como ele deveria funcionar irão surgir. Crie a lista dos requisitos e, uma vez com eles em mão, pode-se especificá-los usando, por exemplo, histórias de usuário.

A partir desse ponto, você pode começar a modelar seu sistema com as técnicas que veremos nas subseções a seguir. Primeiro, inicia-se mapeando as histórias de usuário para diagramas de caso de uso. Depois, identifica-se as principais entidades e os relacionamentos entre elas usando os diagramas de classe.

Por fim, é possível ainda modelar o comportamento dos objetos usando os diagramas de sequência e diagramas de estado.

### Caso de uso

Como já dito, os diagramas de caso de uso são partes da UML para representar a interação entre atores (usuários ou sistemas externos) e um sistema em questão. Ele descreve os principais recursos ou funcionalidades do sistema do ponto de vista do usuário, enfocando as ações que o usuário realiza e as respostas do sistema a essas ações.

Esses diagramas não se concentram nos aspectos técnicos detalhados do sistema, mas sim na funcionalidade e nas interações entre os atores e o sistema. Eles são usados principalmente para capturar os requisitos funcionais do sistema, ou seja, o que o sistema deve fazer em termos de comportamento e funcionalidade.

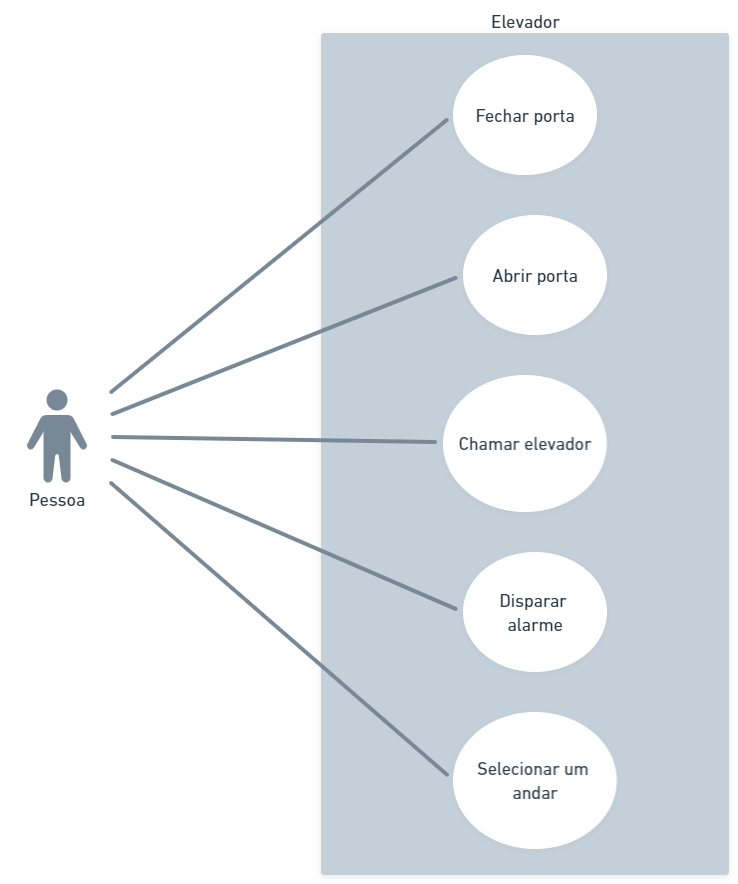
Os diagramas de caso de uso são uma ferramenta de modelagem de alto nível que ajuda a entender e comunicar os recursos principais do sistema de forma clara e compreensível para os stakeholders. Eles são mais úteis para capturar a visão geral do sistema, identificar as principais funcionalidades e estabelecer as interações entre os atores e os casos de uso.

Embora os diagramas de caso de uso não forneçam detalhes técnicos específicos, eles podem ser usados como ponto de partida para outras atividades de engenharia de software. Por exemplo, eles podem servir como base para a elaboração de diagramas de sequência, que descrevem a interação entre os objetos do sistema em um nível mais detalhado. Além disso, os casos de uso identificados podem ser usados para orientar o design e a implementação do sistema.

Portanto, embora os diagramas de caso de uso não abordem diretamente os aspectos técnicos, eles desempenham um papel fundamental na captura de requisitos e na compreensão geral do sistema, fornecendo uma visão funcional que serve como base para atividades posteriores de desenvolvimento e implementação.

Os componentes desse diagrama são: atores, casos de uso e sistema. Veja um exemplo de sua estrutura na figura 11 para um sistema de elevador:

Figura 11 - Diagrama de casos de uso para um elevador



Fonte: Autoria própria, 2023

As pessoas em geral que podem usar o elevador são representadas pelo ator “pessoa”. Esse ator pode executar alguma das funcionalidades descritas em cada um dos casos de uso. O conjunto de funções descrevem um sistema, representado pelo retângulo no fundo.

### Diagrama de classe

Os diagramas de classe são uma ferramenta poderosa na engenharia de software que ajudam a visualizar a estrutura e as relações entre as classes em um sistema orientado a objetos. São parte da UML. Eles representam as classes, seus atributos, métodos e os relacionamentos entre elas. Aqui estão algumas informações sobre os diagramas de classe.

Essa ferramenta é essencial na análise orientada a objetos, que consistem numa abordagem de desenvolvimento de software que se baseia nos princípios da orientação a objetos para analisar, projetar e implementar sistemas. Ela se concentra na modelagem dos objetos do mundo real, suas características (atributos) e comportamentos (métodos), bem como nas relações entre esses objetos.

Os componentes desse diagrama são: nome da classe, atributos e métodos. Veja um exemplo na figura 12:

Figura 12 - Diagrama de classe de uma unidade de manutenção de elevador

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2023

Cada um dos atributos e métodos tem um atributo de visibilidade, representado o encapsulamento daquele recurso. Segue na tabela 1 os diferentes tipos de visibilidade:

Tabela 2 - Visibilidade de atributos e métodos nos diagramas de classe

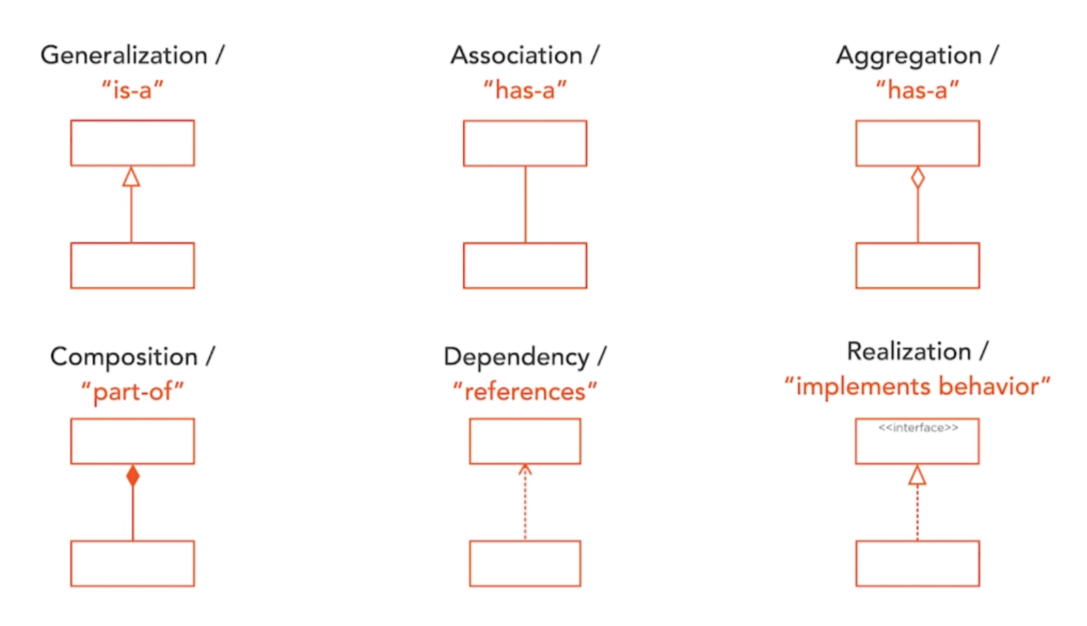
|  |  |
| --- | --- |
| Visibilidade | |
| Tipo | Descrição |
| + publico | Pode ser usado por qualquer outra entidade fora do objeto |
| - privado | Só pode ser usado dentro da própria classe |
| # protegido | Só é acessível dentro da própria classe e das classes filhas |
| ~ *package* | Disponível para todas as classes definidas num pacote |

Fonte: Nyisztor, 2023

Seguindo o princípio "exponha somente o necessário", recomenda-se que se defina atributos e métodos como privados por padrão e, em seguida, forneça métodos públicos para acessar e modificar os atributos. Isso promove o encapsulamento, permitindo que a lógica interna da classe seja controlada e evitando que outros componentes dependam de detalhes de implementação específicos.

Por fim, várias classes podem apresentar diferentes relações entre si. Veja na figura 13 os tipos de relação e sua representação gráfica:

Figura 13 - Tipos de relações entre diagramas de classe e suas representações gráficas



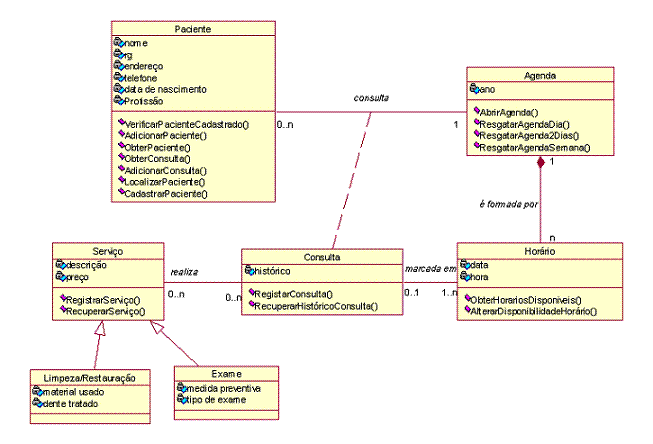
Fonte: Nyisztor, 2023

Segue uma descrição desses relacionamentos:

* Generalização (Herança): O relacionamento de generalização ocorre quando uma classe mais especializada (subclasse) herda características e comportamentos de uma classe mais genérica (superclasse). A subclasse é uma especialização da superclasse, e a generalização é indicada por uma seta sólida que aponta da subclasse para a superclasse;
* Associação: A associação representa um relacionamento estrutural entre duas classes, indicando que objetos de uma classe estão relacionados a objetos de outra classe. Pode ser uma associação unidirecional ou bidirecional. É representada por uma linha entre as classes, e pode conter multiplicidades para indicar a cardinalidade da associação;
* Agregação: A agregação é um relacionamento em que uma classe (todo) contém outras classes (partes), mas as partes podem existir independentemente do todo. É uma associação do tipo "tem-um" ou "tem-vários". É representada por um losango vazado na classe que representa o todo;
* Composição: A composição é um relacionamento mais forte que a agregação, em que uma classe (todo) é composta por outras classes (partes), e as partes não podem existir sem o todo. É uma associação do tipo "tem-um" ou "tem-vários". É representada por um losango preenchido na classe que representa o todo;
* Dependência: A dependência ocorre quando uma classe usa ou depende de outra classe em algum momento, geralmente por meio da chamada de métodos ou referenciando atributos. É um relacionamento temporário e fraco, indicando que uma classe é afetada pelas mudanças em outra classe. É representada por uma linha tracejada com uma seta que aponta para a classe dependente;
* Realização (Implementação): O relacionamento de realização ocorre quando uma classe (classe concreta) implementa uma interface, ou seja, fornece a implementação dos métodos definidos na interface. É usado para mostrar que uma classe está cumprindo um contrato estabelecido por uma interface. É representado por uma linha tracejada com uma seta que aponta para a interface.

Esses relacionamentos podem ocorrer em diferentes níveis de complexidade e em combinação uns com os outros. Na figura 14 é possível ver um diagrama com diferentes relacionamentos:

Figura 14 - Diagrama de classes para um sistema de clínica dentaria



Fonte: Macoratti, 2023?[[10]](#footnote-10)

Os diagramas de classe fornecem uma representação visual desses relacionamentos, facilitando a compreensão da estrutura e das interações entre as classes de um sistema.

Por fim, é possível definir a cardinalidade das relações. No contexto da modelagem de dados, a cardinalidade refere-se à propriedade que descreve a relação entre dois conjuntos de objetos. Em termos mais simples, a cardinalidade representa a quantidade de elementos em um conjunto que podem estar relacionados a elementos de outro conjunto.

A cardinalidade é frequentemente utilizada em diversos modelos de dados, como em diagramas de classe, diagramas entidade-relacionamento (DER) e outros modelos de banco de dados. Ela ajuda a especificar as restrições e as regras de relacionamento entre os elementos.

No contexto dos diagramas de classe, a cardinalidade é usada para descrever a relação entre as classes. Por exemplo, em um relacionamento "um-para-muitos" entre duas classes, a cardinalidade indica quantos objetos da classe A podem se relacionar com quantos objetos da classe B. Pode indicar se a relação é obrigatória ou opcional e se existe um limite superior na quantidade de objetos relacionados.

As cardinalidades mais comuns são expressas através de números ou símbolos como "0", "1", "\*", "n", "m" e outros, conforme mencionado anteriormente. Elas fornecem informações valiosas sobre a estrutura e o comportamento dos objetos em um sistema, permitindo uma melhor compreensão das relações entre eles.

Segue uma lista com algumas cardinalidades comumente usadas:

* Cardinalidades fixas:
  + 0..1: Zero ou um objeto está envolvido no relacionamento;
  + 1: Exatamente um objeto está envolvido no relacionamento;
  + 0.. \*: Zero ou mais objetos estão envolvidos no relacionamento;
  + 1.. \*: Um ou mais objetos estão envolvidos no relacionamento;
  + \*: Qualquer quantidade de objetos pode estar envolvida no relacionamento (zero ou mais).
* Cardinalidades limitadas:
  + n..m: Pelo menos "n" e no máximo "m" objetos estão envolvidos no relacionamento;
  + n.. \*: Pelo menos "n" objetos estão envolvidos no relacionamento (sem limite superior).
* Cardinalidades específicas:
  + 1..1: Exatamente um objeto está envolvido no relacionamento (cardinalidade rígida);
  + 1.. n: Um ou mais objetos estão envolvidos no relacionamento, mas pelo menos um é obrigatório;
  + 0..n: Zero ou mais objetos estão envolvidos no relacionamento;
  + 0.. 1..n: Zero, um ou mais objetos estão envolvidos no relacionamento.

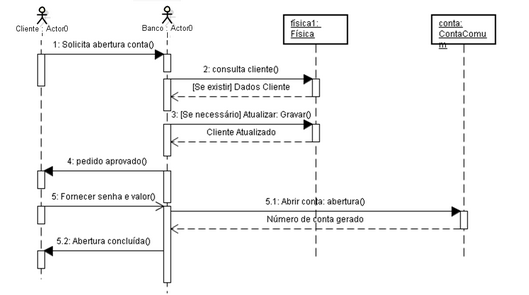
### Diagrama de sequência

Diagramas de sequência são uma forma de representação visual que descreve a interação entre objetos em um sistema ao longo do tempo. Eles mostram a ordem das mensagens trocadas entre os objetos durante a execução de um determinado cenário ou funcionalidade.

Os diagramas de sequência são frequentemente usados para modelar o comportamento dinâmico de um sistema, capturando o fluxo de controle e as trocas de mensagens entre os objetos envolvidos. Eles fornecem uma visão detalhada das interações entre os objetos, permitindo entender como o sistema se comporta em diferentes situações.

A estrutura desse diagrama pode ser vista na figura 15:

Figura 15 - Diagrama de sequência para um sistema bancário

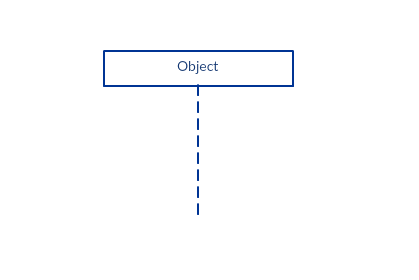


Fonte: Miguel, 2019[[11]](#footnote-11)

Os objetos são representados por retângulos verticais ao longo do eixo vertical do diagrama. Cada objeto é rotulado com seu nome e pode incluir sua instância ou classe correspondente. No exemplo da figura 15, “física1” e “conta” são exemplos de objetos.

As linhas de vida são linhas verticais que se estendem a partir dos objetos e representam a existência do objeto ao longo do tempo. Elas mostram o período em que um objeto está ativo e pode receber ou enviar mensagens. Um diagrama de sequência é composto por várias dessas notações de linha de vida. Duas notações de linha de vida não devem se sobrepor. Eles representam os diferentes objetos ou partes que interagem entre si no sistema durante a sequência. Veja na figura 16 como são representadas as linhas de vida:

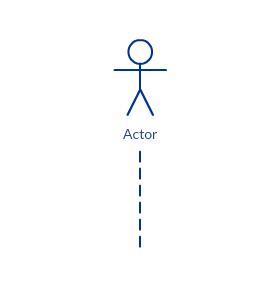
Figura 16 - Linhas de vida no diagrama de sequência



Fonte: Creately, 2021

Uma notação de linha de vida com um símbolo de elemento de ator é usada quando o diagrama de sequência específico pertence a um caso de uso. Veja na figura 17:

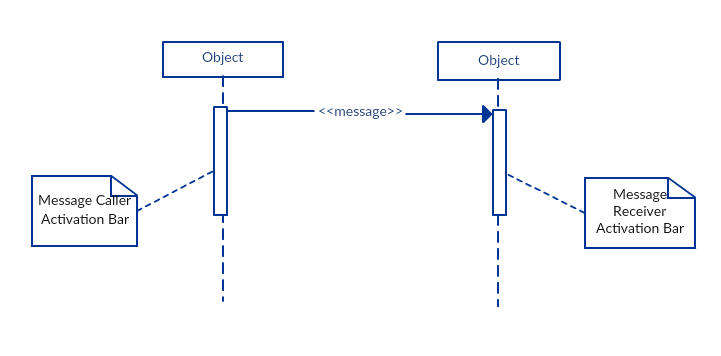
Figura 17 - Atores no diagrama de sequência



Fonte: Creately, 2021

Em um diagrama de sequência, uma interação entre dois objetos ocorre quando um objeto envia uma mensagem para outro. O uso da barra de ativação nas linhas de vida do chamador da mensagem (o objeto que envia a mensagem) e do receptor da mensagem (o objeto que recebe a mensagem) indica que ambos estão ativos/está instanciados durante a troca da mensagem. Veja na figura 18 como são representadas as barras de ativação:

Figura 18 - Barras de ativação no diagrama de sequência

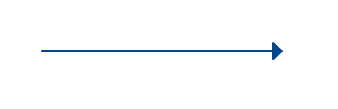


Fonte: Creately, 2021

As mensagens são representadas por setas horizontais entre as linhas de vida dos objetos. Elas indicam a troca de informações ou chamadas de métodos entre os objetos. As mensagens têm diferentes tipos (CREATELY, 2021).

Uma mensagem síncrona (figura 19) é usada quando o remetente espera que o receptor processe a mensagem e retorne antes de continuar com outra mensagem. A ponta de seta usada para indicar este tipo de mensagem é sólida.

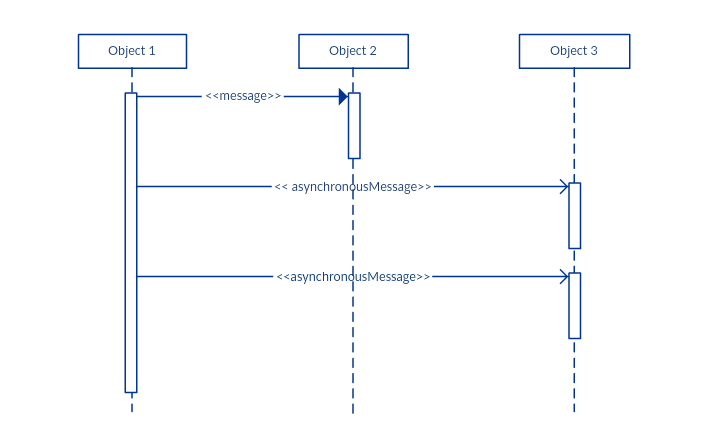
Figura 19 - Seta para representação de mensagem síncrona no diagrama de sequência



Fonte: Creately, 2021

Uma mensagem assíncrona (figura 20) é usada quando o chamador da mensagem não espera que o receptor processe a mensagem e volte antes de enviar outras mensagens para outros objetos dentro do sistema. A ponta de seta usada para mostrar este tipo de mensagem é uma seta de linha.

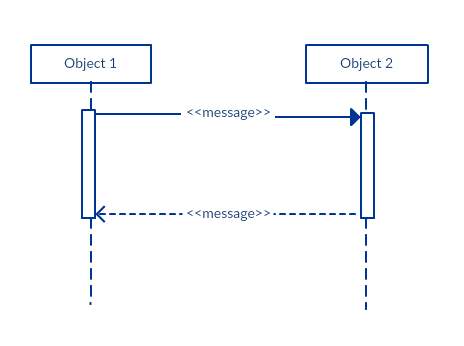
Figura 20 - Seta para representação de mensagem assíncrona no diagrama de sequência



Fonte: Creately, 2021

Uma mensagem de retorno (figura 21) é usada para indicar que o receptor da mensagem terminou o processamento da mensagem e está devolvendo o controle para o autor da chamada da mensagem. As mensagens de retorno são peças opcionais de notação. A barra de ativação, que é acionada por uma mensagem síncrona, implica sempre uma mensagem de retorno.

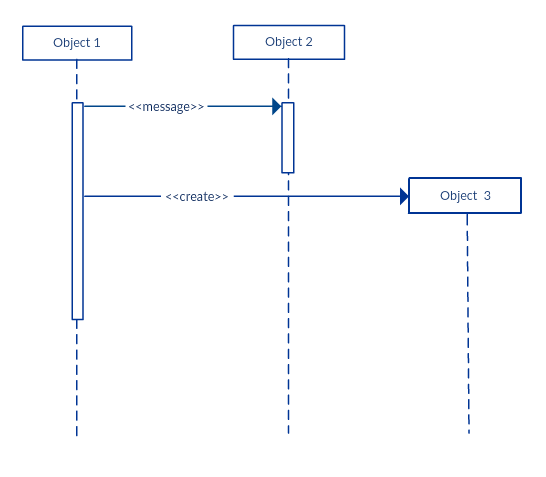
Figura 21 - Seta para representação de uma mensagem de retorno no diagrama de sequência



Fonte: Creately, 2021

Os objetos não vivem necessariamente durante toda a duração da sequência de eventos. Objetos ou participantes podem ser criados de acordo com a mensagem que está sendo enviada (figura 22). A notação “*dropped participant box*” pode ser usada quando você precisa mostrar que o participante em particular não existia até que a chamada de criação seja enviada. Se o participante criado faz algo imediatamente após criação, você deve adicionar uma caixa de ativação logo abaixo da caixa do participante.

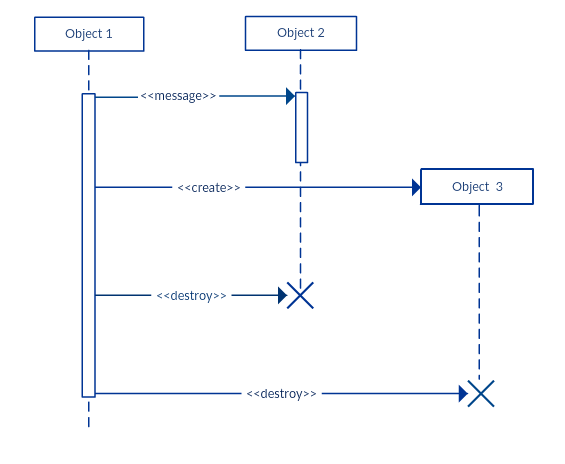
Figura 22 - Seta indicando a criação de um objeto participante no diagrama de sequência



Fonte: Creately, 2021

Da mesma forma, os participantes quando não são mais necessários também podem ser excluídos de um diagrama de sequência. Isto é feito adicionando um “X” no final da linha de vida do referido participante (figura 23).

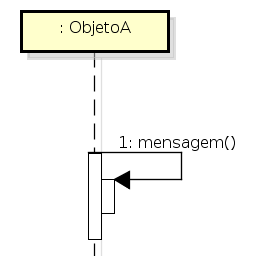
Figura 23 - Um símbolo "X" indicando o fim da participação de um objeto no diagrama de sequência



Fonte: Creately, 2021

Quando um objeto envia uma mensagem para si mesmo, ele é chamado de mensagem reflexiva. É indicado com uma seta de mensagem que começa e termina na mesma linha de vida, como mostrado na figura 24:

Figura 24 - Uma seta que aponta para o próprio objeto no diagrama de sequência



Fonte: Treinamento WAEI/MSE, 2023?[[12]](#footnote-12)

Também, as mensagens podem conter informações adicionais, como parâmetros ou valores de retorno.

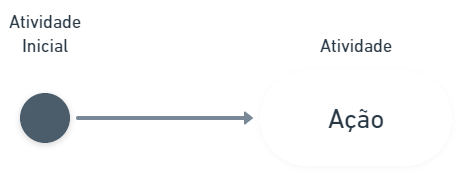
### Diagramas de atividade

O diagrama de atividade é uma ferramenta gráfica utilizada para modelar o fluxo de atividades e a lógica de controle em um processo, sistema ou fluxo de trabalho. Ele descreve a sequência de atividades, as decisões tomadas durante o processo e a ordem em que essas atividades ocorrem.

Os diagramas de atividade são amplamente utilizados na modelagem de processos de negócio, no desenvolvimento de software e em outras áreas que envolvem a representação de fluxos de trabalho. Eles ajudam a visualizar o comportamento e a lógica de um sistema, mostrando as atividades executadas, as decisões tomadas e as ramificações no fluxo de trabalho.

Os dois principais elementos desse diagrama são as atividades e setas (figura 25). As atividades representam ações ou etapas específicas dentro do processo ou fluxo de trabalho. Elas são representadas por retângulos com cantos arredondados e são conectadas por setas que indicam a sequência das atividades.

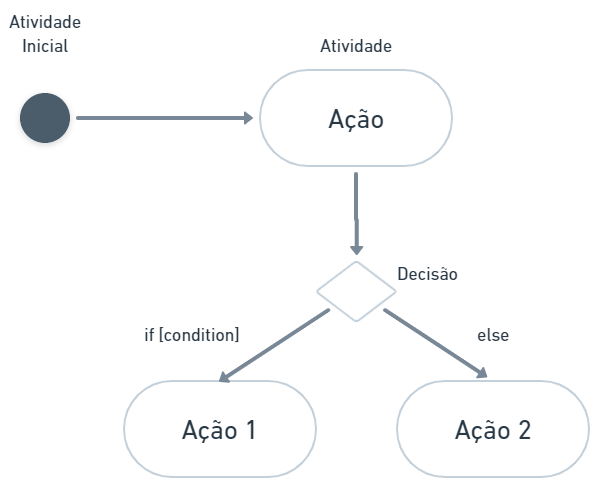
Figura 25 - Atividades e setas no diagrama de atividades



Fonte: Autoria própria, 2023

Ainda, existem as decisões (figura 26), que são pontos no diagrama onde o fluxo de controle se ramifica com base em uma condição ou critério. Elas são representadas por losangos e têm diferentes caminhos de saída, dependendo da condição avaliada. Cada condição é chamada de guarda.

Figura 26 - Símbolo de decisão que ramifica o fluxo de atividades no diagrama de atividades



Fonte: Autoria própria, 2023

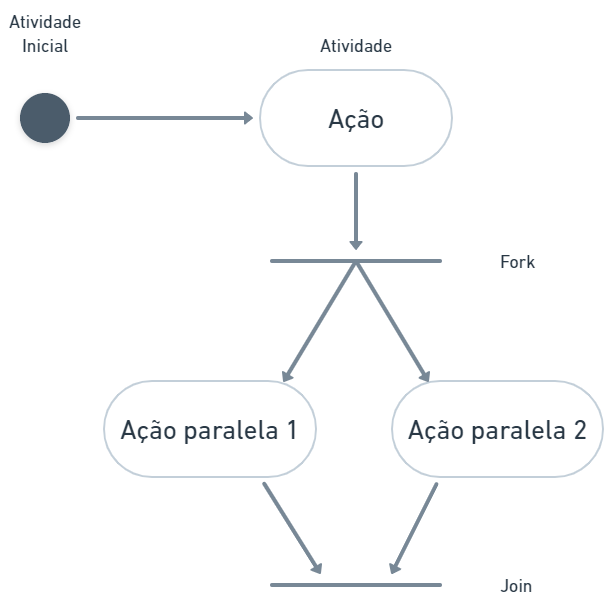
No fim, mais um símbolo de decisão pode ser adicionado para mesclar as atividades novamente em um único fluxo.

Também existem os *forks* e *joins* (figura 27). Forks são pontos no diagrama onde o fluxo de controle se divide em múltiplos caminhos paralelos, indicando atividades simultâneas. Joins são pontos onde os caminhos paralelos se juntam novamente para continuar o fluxo de controle.

Os forks e joins em um diagrama de atividade são adequados para representar a concorrência e a programação paralela em um processo.

Quando um fork é usado em um diagrama de atividade, ele indica um ponto onde o fluxo de controle se divide em vários caminhos paralelos. Cada caminho paralelo representa uma atividade ou ação que pode ser executada simultaneamente por diferentes recursos ou processadores. Isso reflete a ideia de concorrência, onde múltiplas tarefas podem ser realizadas em paralelo.

Figura 27 - Fork e join representando o paralelismo e concorrência no diagrama de atividades



Fonte: Autoria própria, 2023

Por outro lado, o join é usado para indicar o ponto em que os caminhos paralelos se reúnem novamente para continuar o fluxo de controle. O join espera que todos os caminhos paralelos tenham sido concluídos antes de prosseguir para a próxima atividade. Isso garante a sincronização e coordenação adequada entre as atividades paralelas antes de continuar o processo.

Ao usar forks e joins em um diagrama de atividade, é possível modelar a execução concorrente de atividades, capturando a lógica de programação paralela em um processo. Isso é especialmente útil quando se lida com sistemas distribuídos, processamento em tempo real ou qualquer situação em que múltiplas tarefas precisam ser executadas simultaneamente para melhorar a eficiência e o desempenho do sistema.

Os forks e joins fornecem uma representação visual clara das atividades paralelas e ajudam a identificar possíveis problemas de concorrência ou sincronização em um processo. Eles são amplamente utilizados na modelagem de processos empresariais, em sistemas de tempo real e em ambientes de programação paralela para facilitar o entendimento e a análise do fluxo de controle complexo.

### Diagramas de estado

Diagramas de estado, também conhecidos como diagramas de máquina de estado, são ferramentas de modelagem usadas para visualizar o comportamento de um sistema. Eles descrevem as diferentes condições ou estados que um objeto ou sistema pode assumir, bem como as transições entre esses estados em resposta a eventos ou ações específicas.

Os diagramas de estado são amplamente utilizados na engenharia de software para modelar o comportamento de sistemas complexos, como programas de computador ou sistemas embarcados. Eles ajudam a capturar as diferentes condições em que um objeto ou sistema pode existir e a modelar as mudanças de estado em resposta a eventos internos ou externos.

Esses diagramas têm três elementos principais: estados, transições e eventos.

Os estados representam uma condição específica em que um objeto ou sistema pode estar. É representado por um retângulo com o nome do estado.

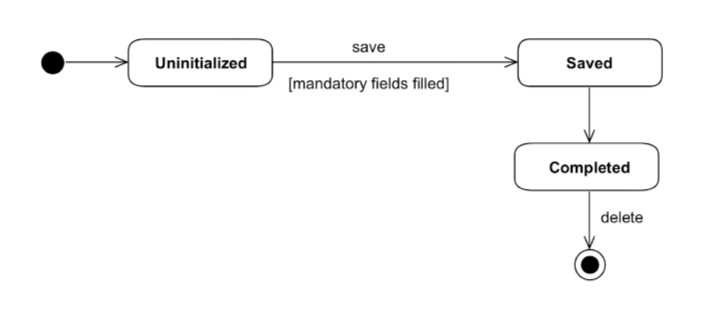
Transições, como o nome diz, representam a mudança de um estado para outro em resposta a um evento ou ação. É representado por uma seta direcionada do estado de origem para o estado de destino, rotulado com o evento ou ação que causa a transição.

Um evento, por sua vez, é um estímulo que desencadeia uma transição de estado. Pode ser qualquer coisa, desde um sinal externo até uma ação interna.

Ainda, existe um elemento opcional, que também estão presente nos diagramas de sequência, que são os *guards*. Os guards, também conhecidos como guardas ou condições de transição, são elementos adicionais em um diagrama de estado que especificam uma condição que deve ser verdadeira para que uma transição de estado ocorra. Eles ajudam a controlar o fluxo de transições, permitindo que apenas certas transições sejam executadas com base nas condições especificadas.

Um exemplo de diagrama de estados pode ser visto na figura 28:

Figura 28 - Diagrama de estados para um sistema de criação de viagens



Fonte: Nyisztor, 2023

O ponto de entrada é indicado pelo círculo preto totalmente preenchido. Isso mostra que se está lidando com um novo objeto criado. O primeiro estado é o “*uninitialized*”. Existe uma transição de estado que leva para um novo estado chamado “*saved*”. Para chegar nesse estado é necessário executar uma ação “save”, que só é realizada se a condição “campos obrigatórios preenchidos” for satisfeita. Por fim, existe o estado “*completed*” e a ação “*delete*” sinaliza a destruição do objeto com um círculo preto semipreenchido.

# DOCUMENTO DE REQUISITO: COMUNICADOR INTER-AGENTES

Versão 1.2.1 – maio de 2023.

## Ficha técnica

### Equipe responsável pela elaboração

Marcos Paulo F. Vaz (Gerente de Projetos)[[13]](#footnote-13).

Matheus dos Santos Silva (Desenvolvedor).

Matheus Oliveira de mores (Desenvolvedor).

Samuel Araujo de Souza (Desenvolvedor)[[14]](#footnote-14).

### Público-alvo

Este manual destina-se a Policiais Militares que trabalham em campo fazendo o patrulhamento e participam, exclusivamente, de operações de cerco e acompanhamento. Também, aos operadores do Centro de Operações da Polícia Militar, que auxiliam policiais de campo nesse tipo de operação. Por fim, aos superiores desses dois grupos, responsável por gerenciá-los em suas rotinas diárias.

## Introdução

Este documento especifica o sistema **Comunicador Inter-agentes**, fornecendo aos desenvolvedores as informações necessárias para o projeto e implementação, assim como para a realização dos testes e homologação do sistema.

Essas informações são extraídas de: polícias que realizam patrulhamento ostensivo (motocicletas e carros); operadores do COPOM; superiores dos agentes.

### Visão geral deste documento

Esta introdução fornece as informações necessárias para fazer um bom uso deste documento, explicitando seus objetivos e as convenções que foram adotadas no texto, além de conter uma lista de referências para outros documentos relacionados. As demais seções apresentam a especificação do sistema Comunicador Inter-agentes e estão organizadas como descrito abaixo.

* **Subseção 4.3** – Descrição geral do sistema: apresenta uma visão geral do sistema, caracterizando qual é o seu escopo e descrevendo seus usuários.
* **Subseção 4.4** – Requisitos funcionais (casos de uso): específica todos os requisitos funcionais do sistema, descrevendo os fluxos de eventos, prioridades, atores, entradas e saídas de cada caso de uso a ser implementado.
* **Subseção 4.5** – Requisitos não funcionais: específica todos os requisitos não funcionais do sistema.
* **Subseção 4.6** – Descrição da interface com o usuário: apresenta desenhos, figuras ou rascunhos de telas do sistema.

### Convenções, termos e abreviações

A correta interpretação deste documento exige o conhecimento de algumas convenções e termos específicos, que são descritos a seguir.

#### Identificação dos requisitos

Por convenção, a referência a requisitos é feita através do identificador do requisito. Por exemplo, o requisito [RF016] é identificado pelo RF (Requisito Funcional) e do número 016. Já o requisito não funcional [NF008] é identificado pelo NF (NãoFuncional) e por 008. Por fim, a interface [IN002] é identificada pelo IN (Interface) e por 002.

#### Prioridade do requisito

Para estabelecer a prioridade dos requisitos foram adotadas as denominações “essencial”, “importante” e “desejável”.

* **Essencial** é o requisito sem o qual o sistema não entra em funcionamento. Requisitos essenciais são requisitos imprescindíveis, que têm que ser implementados impreterivelmente;
* **Importante** é o requisito sem o qual o sistema entra em funcionamento, mas de forma não satisfatória. Requisitos importantes devem ser implementados, mas, se não forem, o sistema poderá ser implantado e usado mesmo assim;
* **Desejável** é o requisito que não compromete as funcionalidades básicas do sistema, isto é, o sistema pode funcionar de forma satisfatória sem ele. Requisitos desejáveis são requisitos que podem ser deixados para versões posteriores do sistema, caso não haja tempo hábil para implementá-los na versão que está sendo especificada.

## Descrição geral do sistema

As operações de cerco e acompanhamento, estabelecidos pela doutrina operacional padrão da Polícia Militar (PM) brasileira, sofrem com um problema identificado a partir de observações e *feedbacks*: modulação manual.

Ao se deparar com um veículo em atividade suspeita que desobedeceu a ordem de parada por parte dos policiais, inicia-se o acompanhamento. É responsabilidade do policial zelar pela manutenção da visibilidade do indivíduo, prestar atenção no trânsito e modular com o Centro de Operações da PM (COPOM) passando as coordenadas e direção do motorista me fuga. A COPOM, por sua vez, repassa as coordenadas para que outros policiais possam fazer o cerco do veículo infrator.

Esse conjunto de responsabilidades pode sobrecarregar o policial e prejudicar sua performance. Também, informações de localização via rádio utilizam apenas um canal da comunicação humana, limitando-se a voz, o que diminui o tempo de reação para a realização do cerco, uma vez que não se sabe a disposição geográfica dos policiais engajados na operação.

O sistema de rastreamento e acompanhamento para policiais é uma solução tecnológica desenvolvida para auxiliar as forças policiais na gestão e monitoramento de ocorrências em tempo real. Os policiais de campo podem utilizar seus smartphones com GPS, em sincronia com uma câmera corporal, para enviar suas coordenadas e características do individuo em fuga de forma automática para o COPOM e outros policiais no apoio, para que as operações sejam mais rápidas e sistematizadas.

Os policiais de campo terão a quantidade de responsabilidades reduzidas, evitando a modulação manual a cada 5 segundos para passar informações de geolocalização. Os operadores da COPOM e policiais no apoio podem ver a fuga em tempo real visualmente num mapa.

### Abrangência e sistemas relacionados

O sistema é desenvolvido em duas plataformas: *mobile* (para policiais que atuam no patrulhamento em campo) e *desktop* (para operadores do COPOM e superiores). Os policiais, quando começarem seu expediente, poderão “iniciar patrulhamento”, isso mudará seu *status* que será visível para o operador do COPOM, que terá uma relação com todos os policiais de campo. A mesma relação estará disponível para superiores dos policiais.

Caso ocorra, o policial poderá “iniciar um acompanhamento” que iniciará uma sessão entre a equipe de policiais em campo com um operador do COPOM – que por sua vez terá a visão da posição geográfica deles num mapa em tempo real.

Tanto a equipe de policiais em campo como o operador do COPOM poderão “solicitar apoio”, que vai enviar notificações para policiais nas proximidades que iniciara o patrulhamento; notificações essas que poderão ser aceitas ou recusadas. Se for aceito, o policial irá ter a disposição geográfica dos policiais no acompanhamento graficamente num mapa em tempo real.

O sistema, portanto, depende de uma conexão com a internet e de uma API para mapas.

Não fazem parte do escopo do sistema:

* Outros tipos de ocorrência: O objetivo é solucionar um problema (modulação manual) específico de ocorrências do tipo “cerco e acompanhamento”. Ocorrências como: som alto, assalto, violência doméstica, etc., não fazem parte do escopo do sistema – a não ser que estes acarretem numa demanda de fuga e acompanhamento;
* Inter aplicabilidade de sistemas: Uso interligado diretamente com outras soluções sistêmicas da polícia, a arquitetura que abrange os requisitos tem como enfoque delimitar a um sistema único que não tenha sua interface funcional ligada a nenhum outro sistema policial;
* Investigação Criminal: O sistema de acompanhamento policial pode registrar e rastrear ocorrências policiais do tipo “cerco e acompanhamento”, mas não inclui funcionalidades para conduzir investigações criminais detalhadas. Isso geralmente é realizado por meio de processos separados e ferramentas especializadas;
* Análise Forense: O sistema de acompanhamento policial não abrange a análise forense de evidências, como análise de DNA, impressões digitais ou balística. Essa atividade é realizada por especialistas forenses em laboratórios especializados;
* Jurisprudência e Processos Legais: Embora o sistema de acompanhamento possa fornecer informações sobre o status de ocorrências, ele não trata da gestão de processos legais ou de atividades relacionadas a jurisprudência, como audiências judiciais, intimações ou documentação jurídica;
* Treinamento Policial: O sistema não é responsável por fornecer treinamento policial. Embora possa incluir recursos de suporte ao treinamento, seu foco principal está no acompanhamento e gestão de ocorrências policiais.

### Descrição dos usuários

1. **Policial Militar de viatura quatro rodas**: Equipe de policial de campo que operam, geralmente, entre 2 e 4 soldados;
2. **Policial Militar de motocicleta**: Dupla de policiais. Sofrem com a modulação manual em operações de cerco e acompanhamento, visto que, em alta velocidade suas vidas são postas em risco;
3. **Centro de comando**: São responsáveis por monitorar as ocorrências em tempo real, coordenar as atividades das equipes no campo, fornece suporte e orientação aos policiais, além de tomar decisões estratégicas com base nas informações fornecidas pelo sistema;
4. **Administradores do Sistema**: São os usuários responsáveis pela administração, configuração e manutenção do sistema. Eles gerenciam permissões de acesso, definem parâmetros do sistema, realizam atualizações e monitoram o desempenho geral do sistema;
5. **Gerentes e Supervisores**: Esses usuários têm acesso ao sistema para obter informações gerenciais, analisar dados estatísticos, gerar relatórios de desempenho, acompanhar indicadores-chave e tomar decisões estratégicas com base nas informações fornecidas pelo sistema;
6. **Equipes de Suporte Técnico**: Podem ser responsáveis por prestar suporte técnico aos usuários, solucionar problemas, realizar atualizações de software, manutenção e garantir a disponibilidade e o desempenho adequado do sistema.

## Requisitos funcionais (casos de uso)

### [RF001] Cadastrar equipe de campo

Este caso de uso descreve como a um gerente pode cadastrar equipes de policiais em campo. Apesar da individualidade de cada soldado, as ocorrências envolvem uma equipe cadastrada.

**Ator:** Superiores dos soldados.

**Prioridade:** Essencial.

**Pré-condições:** O ator deve estar autenticado no sistema e ter a devida autorização para cadastrar as equipes de policiais.

**Pós-condições**: Equipe cadastrada; policiais individuais com acesso ao sistema; nova equipe listada na dashboard de controle.

**Fluxo principal**:

1. O superior das equipes (*admin*) clica no botão “Cadastrar Nova Equipe” numa *dashboard* de controle;
2. O sistema exibe o formulário de criação de equipes;
3. O admin preenche os campos obrigatório para cada membro;
4. O sistema valida os dados em tempo real;
5. O admin clica em “Cadastrar”;

O sistema envia um e-mail para cada policial cadastrado com seu usuário e senha.

### [RF002] Cadastrar operador da COPOM

Este caso de uso descreve como a um gerente pode cadastrar um operador do COPOM.

**Ator:** Superiores dos soldados.

**Prioridade:** Essencial.

**Pré-condições:** O ator deve estar autenticado no sistema e ter a devida autorização para cadastrar os operadores do COPOM.

**Pós-condições**: Operador cadastrado; operador com acesso ao sistema; novo operador listado na dashboard de controle.

**Fluxo principal**:

1. O superior das equipes (*admin*) clica no botão “Cadastrar Novo Operador” numa *dashboard* de controle;
2. O sistema exibe o formulário de criação de operadores;
3. O admin preenche os campos obrigatório;
4. O sistema valida os dados em tempo real;
5. O admin clica em “Cadastrar”;

O sistema envia um e-mail para o operador cadastrado com seu usuário e senha.

### [RF003] Iniciar operações

Este caso de uso descreve como um membro da COPOM pode iniciar seu expediente.

**Ator:** Operador do COPOM.

**Prioridade:** Essencial.

**Pré-condições**: Ter sido cadastrado no sistema; fazer autenticação.

**Pós-condições**: Dashboard de controle do operador.

**Fluxo principal**:

1. O operador faz a autenticação no sistema;
2. O sistema apresenta uma dashboard para o operador;
3. O operador clica em “Iniciar Operações”;
4. O sistema atualiza o status do operador para “em patrulhamento”;
5. O sistema atualiza a dashboard de controle dos superiores;
6. O sistema disponibiliza o operador para pegar ocorrências.

### [RF004] Iniciar patrulhamento

Este caso de uso descreve como a equipe policial iniciar seu expediente.

**Ator:** Equipe de policiais.

**Prioridade:** Essencial.

**Pré-condições**: Ter sido cadastrado no sistema; fazer autenticação.

**Pós-condições**: Dashboard de controle do patrulheiro.

**Fluxo principal**:

1. O policial faz a autenticação no sistema;
2. O sistema apresenta uma dashboard para o agente;
3. O policial clica em “Iniciar Patrulhamento”;
4. O sistema sincroniza o início de patrulhamento em todos os membros da equipe individualmente – ou seja, o estado deles passa a ser “em patrulhamento”;
5. O sistema atualiza a dashboard de controle dos superiores;
6. A equipe fica disponível para prestar apoio ou iniciar uma ocorrência.

### [RF005] Iniciar acompanhamento

Este caso de uso descreve como a equipe policial em patrulhamento pode iniciar uma sincronização com o COPOM para evitar a modulação manual.

**Ator:** Equipe de policiais.

**Prioridade:** Essencial.

**Pré-condições:** Ter iniciado o patrulhamento (expediente de trabalho).

**Pós-condições**: Dashboard de acompanhamento.

**Fluxo principal**:

1. O policial clica no botão “Iniciar Acompanhamento”;
2. O sistema liga a câmera do policial;
3. O sistema cria um *socket* entre o policial e o operador do COPOM;
4. O sistema abre um mapa e atualiza as coordenadas do policial em tempo real;
5. O sistema envia coordenadas para o COPOM e atualiza o mapa deles em tempo real.

### [RF006] Solicitar apoio

Este caso de uso descreve como a equipe policial em um acompanhamento pode solicitar apoio para o cerco.

**Ator:** Equipe de policiais.

**Prioridade:** Essencial.

**Pré-condições:** Ter iniciado um acompanhamento (ocorrência).

**Pós-condições**: Dashboard de acompanhamento.

**Fluxo principal**:

1. O policial clica no botão “Solicitar Apoio”;
2. O sistema faz uma varredura de policiais na região que iniciaram o patrulhamento e não estão em nenhuma ocorrência;
3. Os policiais em patrulhamento podem “aceitar” ou “recusar” a solicitação de apoio;
4. O sistema adicionar os policiais que aceitaram o apoio na ocorrência;
5. O sistema sincroniza a posição dos policiais que aceitaram o apoio no mapa.

### [RF007] Acompanhar livestream

Este caso de uso descreve como um operador da COPOM pode abrir, em tempo real, a câmera de um policial que iniciou um acompanhamento (como uma *live stream*). O objetivo é evitar que o policial module manualmente informações sobre as características do veículo e infrator.

**Ator:** Operador do COPOM.

**Prioridade:** Desejável.

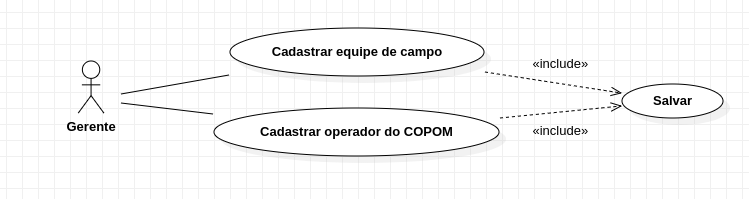
**Pré-condições:** Equipe policial ter iniciado um acompanhamento (ocorrência).

**Pós-condições**: Lista com todos os policiais na ocorrência; interface de live stream.

**Fluxo principal**:

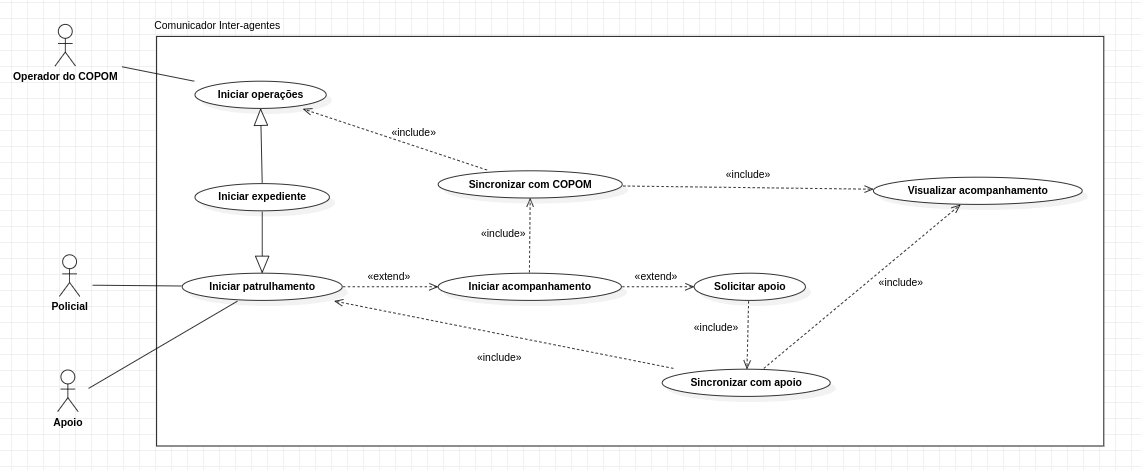
1. O operador clica o botão “Assistir” no soldado desejado;
2. Uma interface com o vídeo da câmera do policial é aberta.

Figura 29 - Caso de uso para cadastros de equipes de policiais e operadores do COPOM



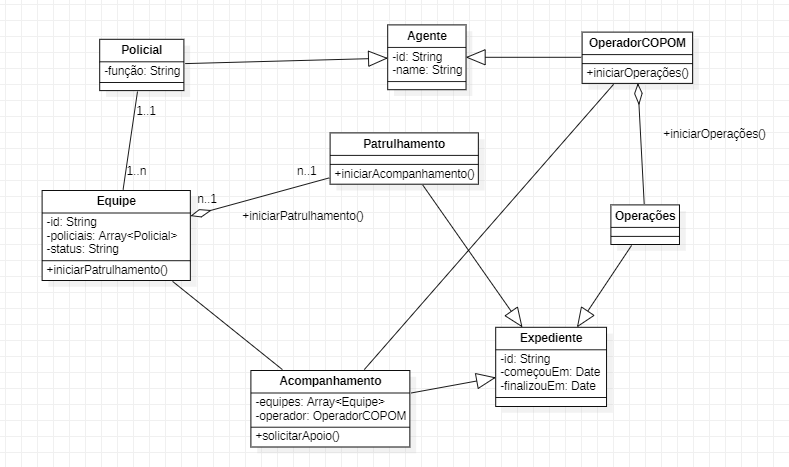
Fonte: Autoria própria, 2023

Figura 30 - Casos de uso para o sistema Comunicador Inter-agentes



Fonte: Autoria própria, 2023

Figura 31 - Diagrama de classe do Comunicador Inter-agente



Fonte: Autoria própria, 2023

## Requisitos não funcionais

### [NF001] Interface de patrulhamento minimalista

Quando em patrulhamento, os policiais devem manter todo o seu foco no trânsito e em eventuais suspeitos. Quanto menor a quantidade de “clicks” e informações na tela, melhor para o policial.

**Prioridade**: Essencial.

**Casos(s) de uso associado(s)**: [RF004]; [RF005].

### [NF002] Botão para acionar “iniciar acompanhamento”

Quando um indivíduo desobedecer a ordem de parada e empreender fuga, o policial deve com facilidade ser capaz de clicar em um botão para “iniciar o acompanhamento”. O correto acionamento deve ser respondido com um feedback claro, para que não haja dúvidas de que ele conseguiu iniciar o acompanhamento e uma sincronização com o COPOM está sendo feita. Por fim, erros de clique no botão devem ser minimizados.

**Prioridade**: Importante.

**Casos(s) de uso associado(s)**: [RF005].

### [NF003] Usar voz para acionar “iniciar acompanhamento”

Quando um indivíduo desobedecer a ordem de parada e empreender fuga, pensando nos policiais que trabalham em motocicletas principalmente, seria interessante que eles não tirassem as mãos do guidão da moto. Para isso, poderia ser possível acionar o “iniciar o acompanhamento” com a voz.

**Prioridade**: Desejável.

**Casos(s) de uso associado(s)**: [RF005].

### [NF005] Utilização do 5G para comunicação em tempo real

O sistema deve ser capaz de aproveitar a tecnologia de comunicação 5G para fornecer uma conectividade de alta velocidade, baixa latência e confiável entre os policiais envolvidos na perseguição, operadores do COPOM e outros policiais na região. A utilização do 5G garantirá uma transmissão rápida e eficiente das coordenadas e informações gráficas no mapa, permitindo uma colaboração ágil e precisa entre as equipes.

**Prioridade**: Importante.

**Casos(s) de uso associado(s)**: [RF005]; [RF006]; [RF007].

### [NF006] Manter a sessão em regiões sem cobertura de sinal

O sistema deve ser resiliente e, ao adentrar zonas com baixa disponibilidade de sinal, deve ser capaz de continuar armazenando as coordenadas para enviar para os demais assim que possível.

**Prioridade**: Essencial.

**Casos(s) de uso associado(s)**: [RF005]; [RF006].

### [NF007] Segurança e prevenção de interceptação de mensagens

O sistema deve ser projetado com medidas robustas de segurança para garantir a confidencialidade, integridade e autenticidade das mensagens transmitidas entre os policiais envolvidos na perseguição, operadores do COPOM e outros policiais na região. Deve-se implementar mecanismos adequados de criptografia, autenticação e controle de acesso para prevenir a interceptação ou acesso não autorizado às informações sensíveis transmitidas.

A segurança da comunicação é de extrema importância para garantir que as informações transmitidas durante uma perseguição sejam acessíveis apenas pelas partes autorizadas. A interceptação de mensagens por indivíduos não autorizados pode comprometer a eficácia da operação, colocar em risco a segurança dos envolvidos e prejudicar a integridade das investigações policiais. Portanto, é essencial implementar medidas de segurança adequadas para proteger a confidencialidade e a privacidade dos dados transmitidos.

**Prioridade**: Essencial.

**Casos(s) de uso associado(s)**: [RF005]; [RF006].

### [NF008] Quantidade de policiais no apoio

Muitos policiais podem aceitar o apoio solicitado. O sistema deve ser capaz de lidar com um aumento no número de usuários, dispositivos e mensagens transmitidas, garantindo um desempenho consistente e sem degradação significativa em situações de alto tráfego.

**Prioridade**: Essencial.

**Casos(s) de uso associado(s)**: [RF006].

### [NF009] Baixo tempo de resposta

O sistema deve fornecer uma resposta rápida e eficiente, garantindo tempos de resposta baixos e desempenho otimizado, mesmo em condições de carga intensa ou conexões de rede mais lentas. Porque, assim, o tempo de reação dos policiais no apoio e dos operadores do COPOM será dentro da realidade, melhorando a coordenação e estratégia dos policiais na operação.

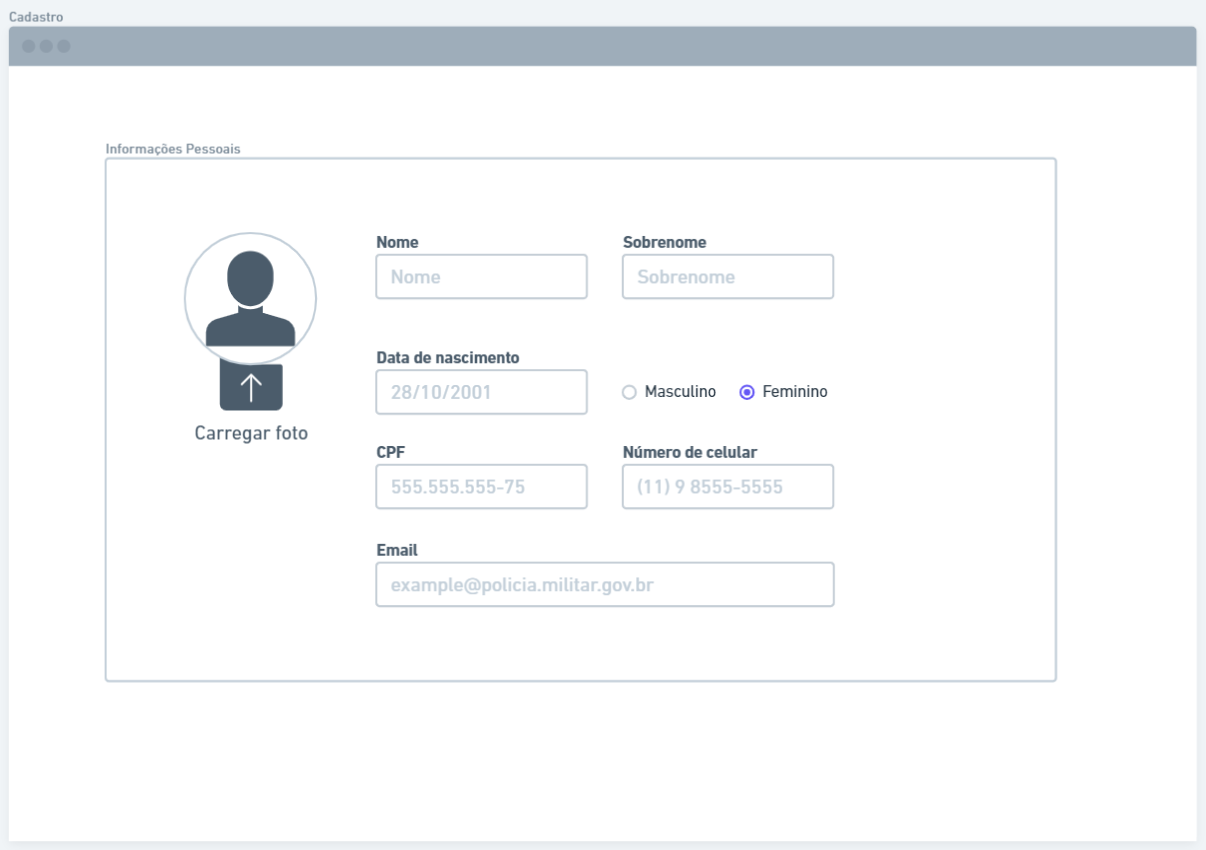
**Prioridade**: Essencial.

**Casos(s) de uso associado(s)**: [RF006]; [RF007].

## Descrição da interface com o usuário

### [IN001] Cadastro de equipes e operadores do COPOM

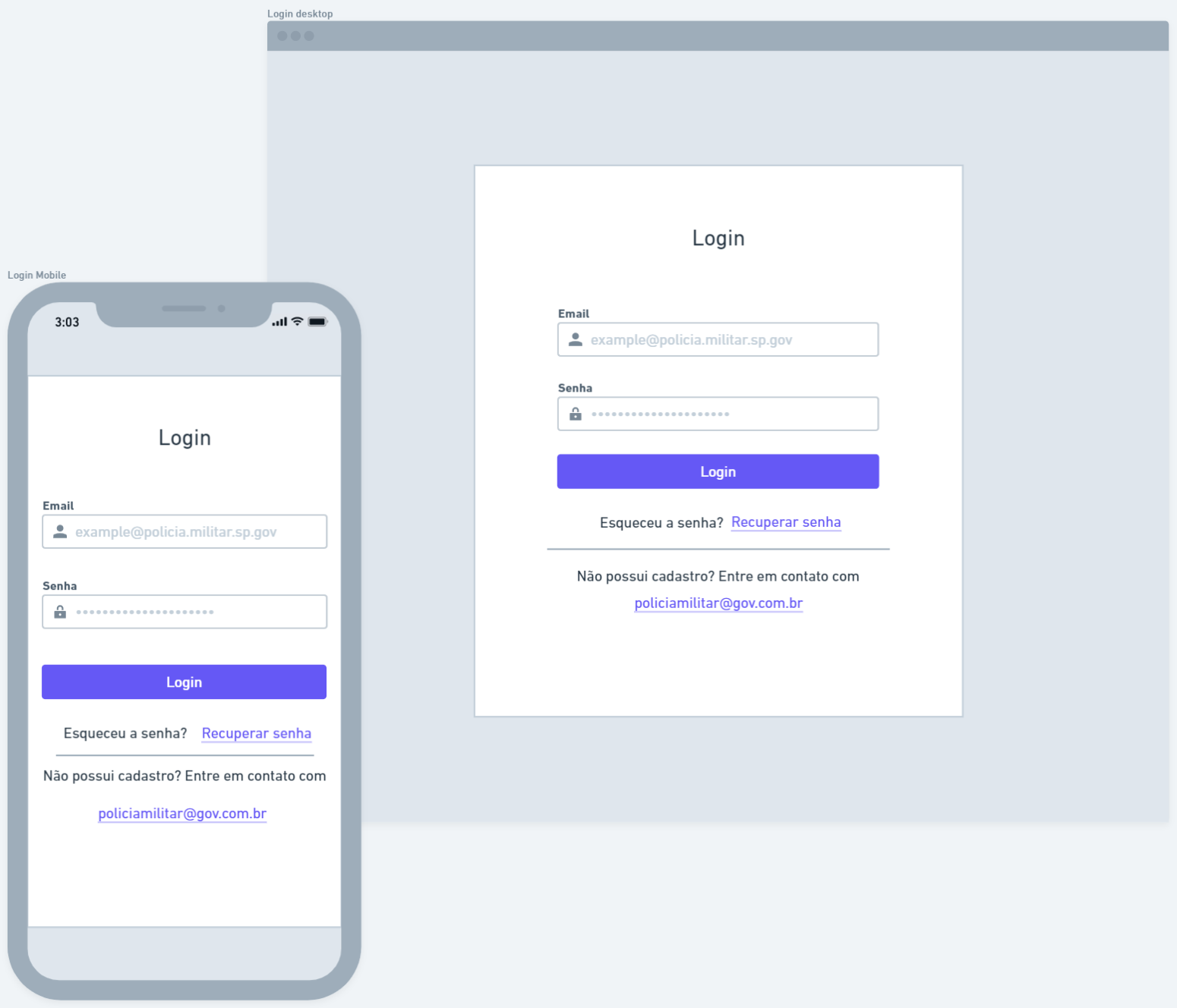
Figura 32 - Rascunho da interface de cadastro de policiais e operadores do COPOM



Fonte: Autoria própria, 2023

### [IN002] Login mobile e desktop

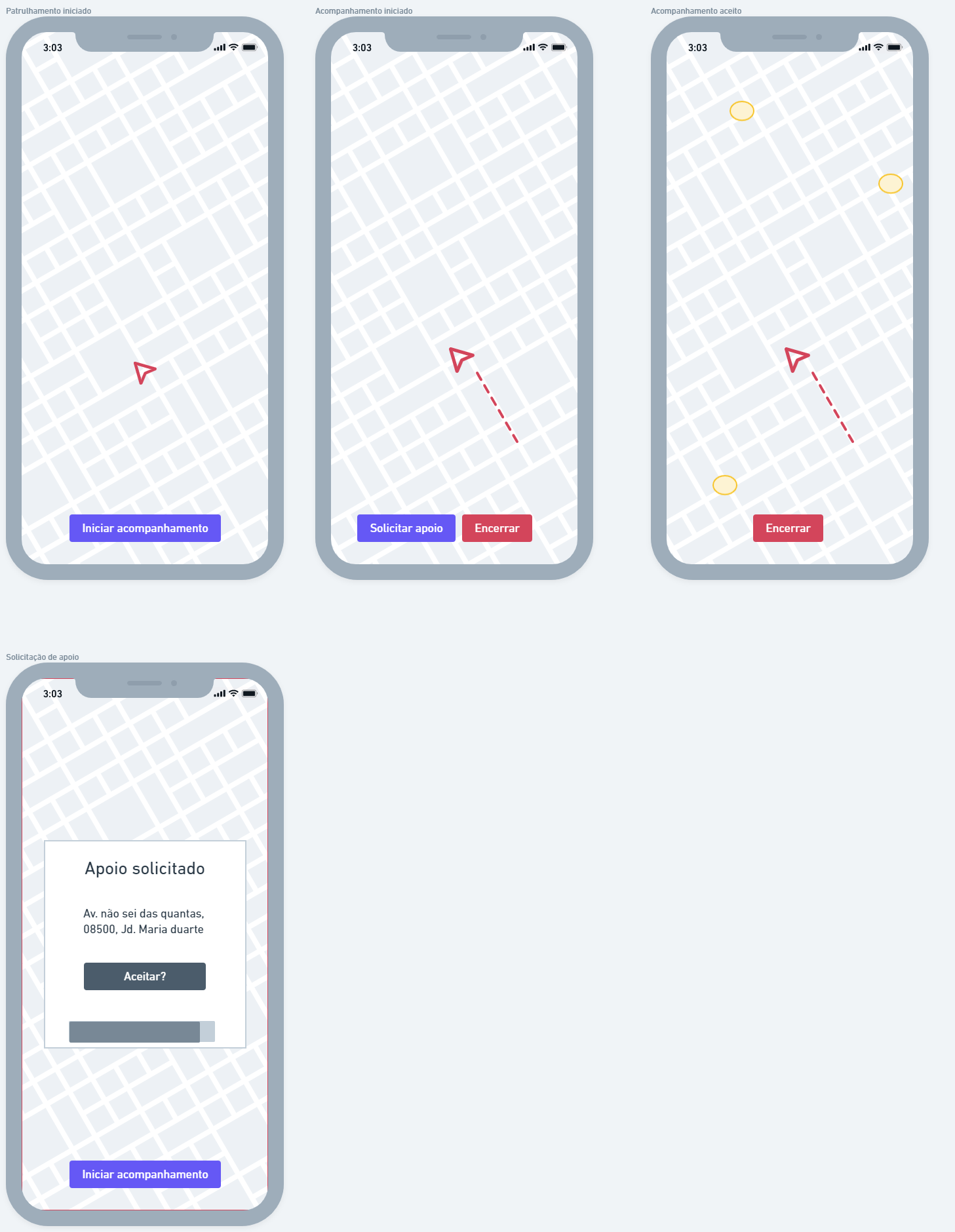
Figura 33 – Rascunho da interface de login desktop e mobile



Fonte: Autoria própria, 2023

### [IN003] Inicio de patrulhamento e solicitação de apoio

Figura 34 - Telas de patrulhamento, início de acompanhamento e solicitação de apoio



Fonte: Autoria prória, 2023

# CONCLUSÃO

Na introdução foi definida uma situação problema: requisitos definidos que não satisfizeram o cliente no momento de sua entrega. Foi visto que é necessário um procedimento bem definido para que situações assim não ocorram. Por sorte, existe uma área que é resultado do acúmulo de muito conhecimento ao longo do tempo, de vários projetos, que documenta boas práticas para execução: a engenharia de requisitos.

Os clientes sabem muito bem o que querem, só não sabem explicar direito. Por esse motivo, foram desenvolvidas técnicas de elicitação de requisitos, isto é, descobrir o que faz parte do escopo do projeto. O intuito é minimizar a quantidade de erros possíveis, pois estes geram custos, e aumentar a satisfação. Para este fim, também foram desenvolvidas técnicas de validação de requisitos, que servem para garantir que os requisitos são realmente úteis para um determinado objetivo.

Ainda, sobre os requisitos, existe uma fase de análise, onde são agrupados e priorizados os requisitos de negócio.

Por fim, foi visto que modelar graficamente os requisitos é útil em praticamente todos os processos da engenharia de requisitos. A modelagem ajuda na comunicação entre os desenvolvedores e, também, ajuda na identificação e elicitação de requisitos, uma vez que o raciocínio fica muito mais claro. O desenvolvimento de código fica muito mais simples, uma vez que toda sua estrutura foi previamente pensada.

Com engenheiros de requisitos surgindo ao redor do mundo, e muitos projetos sendo desenvolvidos, foi criado uma linguagem de modelagem padrão, que serve como um protocolo entre todos os engenheiros; contribuindo para manutenção e desenvolvimento. Essa linguagem é conhecida como *Unified Modeling Language* (UML) e é subdividida em vários tipos de diagramas, cada um com seu propósito. Podemos citar: diagramas de caso de uso; diagramas de classe; diagramas de sequência e diagramas de estado.

O conhecimento dos procedimentos mais comuns da engenharia de requisitos, e da modelagem via UML, foi de suma importância para a elaboração de um documento de requisitos para o sistema Comunicador Inter-agentes. A visão ficou muito mais clara entre os colaboradores e cliente. Além disso, teve-se plena certeza que os requisitos são válidos tendo em vista os objetivos do projeto.

# REFERÊNCIAS

CARDOSO, B. “O cliente não sabe o que quer!”. Será? **O dono do produtoo**, 2020. Disponivel em: <https://odonodoproduto.com/o-cliente-nao-sabe-o-que-quer-sera/>. Acesso em: 20 Maio 2023.

CONCEITO.DE (ed.). **Requisito**. [*S. l.*], [2023?]. Disponível em: https://conceito.de/requisito. Acesso em: 1 abr. 2023.

CREATELY. Tutorial do Diagrama de Sequência: Guia completo com exemplos. **Creately**, 2021. Disponivel em: <https://creately.com/blog/pt/diagrama/tutorial-do-diagrama-de-sequencia/>. Acesso em: 23 Maio 2023.

FREITAS, Paulo. **Engenharia de requisitos**. São Paulo, 08 fev. 2019. Apresentação do Power Point. Para acesso ao conteúdo, contate o autor: https://www.linkedin.com/in/paulo-freitas-7a41b4/.

FREITAS, Paulo. **Gestão de escopo**. São Paulo, 18 ago. 2022. Apresentação do Power Point. Para acesso ao conteúdo, contate o autor: https://www.linkedin.com/in/paulo-freitas-7a41b4/.

G. SUTCLIFFE, Alistair. **Requirements Engineering**. [*S. l.*], 11 abr. 2023. Disponível em: https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/requirements-engineering. Acesso em: 1 abr. 2023.

IRON Man. Direção: Jon Favreau. Intérprete: Robert Downey Jr. Roteiro: Mark Fergus, Hawk Ostby, Art Marcum. [*S. l.*: *s. n.*], 2008. Disponível em: https://www.disneyplus.com/movies/iron-man/6aM2a8mZATiu. Acesso em: 21 maio 2023.

L. ANDERSON, John. **President's perspective: what is engineering?** [*S. l.*], 18 out. 2019. Disponível em: https://www.nae.edu/221278/Presidents-Perspective-What-Is-Engineering. Acesso em: 1 abr. 2023.

NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS. Technology & Engineering Literacy Framework: for the 2018 National Assessment of Educational Progress. **The National Assessment Governing Board**, Washington, DC, p. 16, 2018.

NYISZTOR, Károly. **UML and Object-Oriented Design Foundations**. Udemy. 2023. Disponível em: https://www.udemy.com/course/uml-and-object-oriented-design-foundations/. Acesso em: 23 de mai. 2023.

PROJECT MANAGEMENT INSTITUTE. **Um guia do conhecimento em Gerenciamento de Projetos (Guia Pmbok)**. 5ª. ed. [*S. l.*]: Project Management Institute, 2013. 589 p. ISBN 1628250070.

SUTHERLAND, J. **Scrum:** a arte de fazer o dobro do trabalho na metade do tempo. São Paulo: Leya, 2014.

W. SINN, John; ERION, John; UWAKWEH, Benjamin; SCHULT, Thomas; LAHIDJI, Robert. Industrial Technology Programs: Relationship to Engineering and Management. **Administrative Science**, [*S. l.*], ano 1990, v. 16, n. 1, p. 5-9. Disponível em: https://www.jstor.org/stable/43603003. Acesso em: 1 abr. 2023.

Tabela

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente com confiança média

1. Disponível em: https://www.standishgroup.com/sample\_research\_files/CHAOSReport2015-Final.pdf. Acessado em: 21 de maio de 2023 [↑](#footnote-ref-1)
2. Disponível em: https://protocolosmarvel.wordpress.com/2023/03/20/howard-stark/3/. Acessado em: 21 de maio de 2023 [↑](#footnote-ref-2)
3. Disponível em: https://www.legiaodosherois.com.br/lista/10-coisas-que-voce-precisa-saber-sobre-o-reator-arc.html. Acessado em: 21 de maio de 2023 [↑](#footnote-ref-3)
4. Disponível em: https://www.zendesk.com.br/blog/perguntas-para-recrutar-vendedores/. Acessado em: 21 de maio de 2023 [↑](#footnote-ref-4)
5. Disponível em: https://www.volusion.com/blog/5-tips-to-actually-make-group-brainstorming-effective/. Acessado em: 22 de maio de 2023 [↑](#footnote-ref-5)
6. Disponível em: https://youtu.be/Y1x5JeSsicw. Acessado em: 21 de maio de 2023 [↑](#footnote-ref-6)
7. Disponível em: https://www.sketch.com/blog/wireframe-examples/. Acessado em: 22 de maio de 2023 [↑](#footnote-ref-7)
8. Disponível em: https://www.cnet.com/tech/computing/figma-design-tool-gets-internet-scale-sharing-and-community/. Acessado em: 22 de maio de 2023 [↑](#footnote-ref-8)
9. Disponível em: https://www.sitepoint.com/prototyping-with-adobe-xd/. Acessado em: 22 de maio de 2023 [↑](#footnote-ref-9)
10. Disponível em: https://www.macoratti.net/net\_uml1.htm. Acessado em: 23 de maio de 2023 [↑](#footnote-ref-10)
11. Disponível em: https://medium.com/documenta%C3%A7ao-uml/introdu%C3%A7%C3%A3o-ao-diagrama-de-sequ%C3%AAncia-1ea5e9563594. Acesso em: 23 de maio de 2023 [↑](#footnote-ref-11)
12. Disponível em: https://treinamentowaei.wordpress.com/diagrama-de-sequencia/. Acessado em: 23 de maio de 2023 [↑](#footnote-ref-12)
13. Dúvidas, críticas e sugestões devem ser encaminhadas por escrito para o seguinte endereço eletrônico: [marcospaulovaz@hotmail.com](mailto:marcospaulovaz@hotmail.com). [↑](#footnote-ref-13)
14. Dúvidas, críticas e sugestões devem ser encaminhadas por escrito para o seguinte endereço eletrônico: [samuel.araujo.souza@outlook.com](mailto:samuel.araujo.souza@outlook.com).

    Recomendamos que o assunto seja identificado com o título desta obra. Alertamos ainda para a importância de se identificar o endereço e o nome completos do remetente para que seja possível o envio de respostas. [↑](#footnote-ref-14)